REMARKS

Introduction

Claims 1-15 are pending in this application, of which claims 1, 5 and 7-11 being independent. Claims 5, 6, 8, 10 and 15 have been withdrawn from consideration for prosecution of the present application. Claims 1, 3, 7, 9 and 11-14 have been amended to clarify aspects of the present application. Amendments to the claims are supported on page 11, line 18 – page 12, line 3; page 15, line 24 – page 16, line 24; and Figure 5 of the present application. Care has been exercised not to introduce new matter.

Information Disclosure Statement

In the instant Office Action, the Examiner asserts that the information disclosure statement filed September 4, 2008 fails to comply with 37 C.F.R. 1.98(a)(2) because it does not include a legible copy of each cited foreign patent document. Applicants respectfully request that JP 2000-148783, JP 2000-298637, JP 2001-160024 and JP 2001-209568 be considered because Applicants note that copies with English abstracts of JP 2000-148783, JP 2000-298637, JP 2001-160024 and JP 2001-209568 were submitted on September 4, 2008. Accordingly, Applicants respectfully request that the Examiner consider the above identified information disclosure statement and return the initialed copy of form PTO-1449 in the next official action.

Further, Applicants respectfully request that the Examiner consider "X/Open CAE Specification" listed on the information disclosure statement filed February 17, 2004 and return the initialed copy of the form PTO-1449 in the next official action.

For the Examiner's convenience, Applicants are submitting the best available copies of the above documents together with this document.

Claim Rejections – 35 U.S.C. § 103

Claims 1-2, 7, 9 and 11-12 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard (U.S. 5,729,689) and further in view of Sitaraman (US 6,427,170). Claims 3 and 13 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard and Sitaraman as applied to claims 1 and 11, and further in view of BenShaul (US Pub. 2002/0010798). Claims 4 and 14 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard, Sitaraman and BenShaul as applied to claims 3 and 13, and further in view of Beser (US 6,754,622). Applicants respectfully traverse these rejections for at least the following reasons.

Applicants respectfully submit that none of the cited references discloses or suggests the features as recited by amended claims 1, 7, 9 and 11, specifically "each of a plurality of nodes having system definition for other nodes." In the claimed subject matter, a name information retrieval request is arranged to be transmitted to each of the plurality of nodes respectively, each of the nodes having its system definition (see, for example, at page 15, line 24 to page 16, line 24 of the specification). Then, the name information for each of the nodes is acquired.

Applicants respectfully submit that none of the cited references discloses the above discussed limitations. The Examiner asserts that Allard discloses a [third] node designated by a system definition at col. 16, lines 4-15 of Allard. However, Allard fails to disclose that the node is designated **by the system definition**. Further, Sitaraman appears to disclose a transmission operation of retrieval request toward each of the plurality cache memories. However, Sitaraman fails to disclose that each of the plurality of nodes has a system definition. In addition, other cited reference also fails to disclose the above discussed limitations of claims 1, 7, 9 and 11.

As such, it is clear that none of the cited references, taken alone or in any combination thereof, renders claims 1, 7, 9 and 11 or any claims dependent thereupon obvious. Applicants

respectfully request that the Examiner withdraw the rejection of claims 1-4, 7, 9 and 11-14 under 35 U.S.C. § 103(a).

CONCLUSION

Having fully responded to all matters raised in the Office Action, Applicant submits that all claims are in condition for allowance, an indication for which is respectfully solicited. If there are any outstanding issues that might be resolved by an interview or an Examiner's amendment, the Examiner is requested to call Applicant's attorney at the telephone number shown below.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP

Takashi Saito
Limited Recognition No. L0123

Please recognize our Customer No. 20277 as our correspondence address.

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 TS:MaM
Facsimile: 202.756.8087
Date: February 25, 2009

Docket No.: 062807-0146 COPY **PATENT**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of	:	Customer Number: 20277
Makoto KOIKE, et al.	:	Confirmation Number: 5912
Application No.: 10/696,563	:	Group Art Unit: 2152
Filed: October 30, 2003	:	Examiner: J. Chang

For: SYSTEM AND METHOD FOR PROVIDING NAMING SERVICE IN DISTRIBUTED PROCESSING SYSTEM

INFORMATION DISCLOSURE STATEMENT

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In accordance with the provisions of 37 C.F.R. 1.56, 1.97 and 1.98, the attention of the Patent and Trademark Office is hereby directed to the references listed on the attached form PTO-1449. It is respectfully requested that the references be expressly considered during the prosecution of this application, and that the references be made of record therein and appear among the "References Cited" on any patent to issue therefrom.

This Information Disclosure Statement is being filed within three months of the U.S. filing date OR before the mailing date of a first Office Action on the merits. No certification or fee is required.

The following are brief comments on the subject matter of the five listed items.

1) Item 1 discloses a technique regarding a so-called wide area load dispersion system and method for performing load dispersion and failure avoidance between each of the pair of servers respectively provided within each of the networks different from each other.

2) Item 2 discloses a name resolver for resolving a name of host for a space including a plurality of domain names.

3) Item 3 discloses a management selecting method for a server application by which a client application is arranged to appropriately select an available server, and to connect it to a network.

4) Item 4 discloses an apparatus and a method for a directory service by which processing load can be made reduced when the whole system is synchronized.

5) With respect to Item 5, descriptions in paragraph [0019] can be read as "Fig. 3 shows one example of an arrangement of a transmitter-side directory server 101. In a directory memory unit 302, directories with structures shown in Fig. 2 can be stored. A directory operating unit 301 is arranged to add data into the directory memory unit 302, and to delete data having been stored in the directory memory unit 302. Information on an updating operation performed by the directory operating unit 301 can be supplied to an updated information generation unit 303. At the updated information generation unit 303, updated information can be generated in response to an updating operation, which updated information is used for copying the updated data of a directory to another directory having been provided within a pair of receiver-side directory servers 102 and 102".

Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP

Brian K. Just Reg. No. 51,321

to Keith E. George
Registration No. 34,111

Please recognize our Customer No. 20277 as our correspondence address.

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 KEG:mjb
Facsimile: 202.756.8087
Date: September 3, 2008

COPY

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Substitute for form 1449/PTO)

ATTY. DOCKET NO.	SERIAL NO.
062807-0146	10/696,563
APPLICANT: Makoto KOIKE, et al.	
FILING DATE: October 30, 2003	GROUP: 2152

U.S. PATENT DOCUMENTS

EXAMINER'S INITIALS	CITE NO.	Document Number Number-Kind Code2 (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US			
		US			
		US			

FOREIGN PATENT DOCUMENTS

EXAMINER'S INITIALS	CITE NO.	Foreign Patent Document Country Codes3-Number 4-Kind Codes (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines Where Relevant Figures Appear	Translation Yes	Translation No
	1	JP-A-2001-092766	04/06/2001	HITACHI LTD		JAPAN (w/ English abstract)	
	2	JP-A-2000-148783	05/30/2000	NEC CORP		JAPAN (w/ English abstract)	
	3	JP-A-2000-298637	10/24/2000	KYUSYU-NEC		JAPAN (w/ English abstract)	
	4	JP-A-2001-160024	06/12/2001	NEC CORP		JAPAN (w/ English abstract)	
	5	JP-A-2001-209568	08/03/2001	SONY CORP		JAPAN (w/ English abstract)	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

EXAMINER'S INITIALS	CITE NO.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	

EXAMINER	DATE CONSIDERED

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

1 Applicant's unique citation designation number (optional). 2 Applicant is to place a check mark here if English language Translation is attached.

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2000-298637**
(43)Date of publication of application : **24.10.2000**

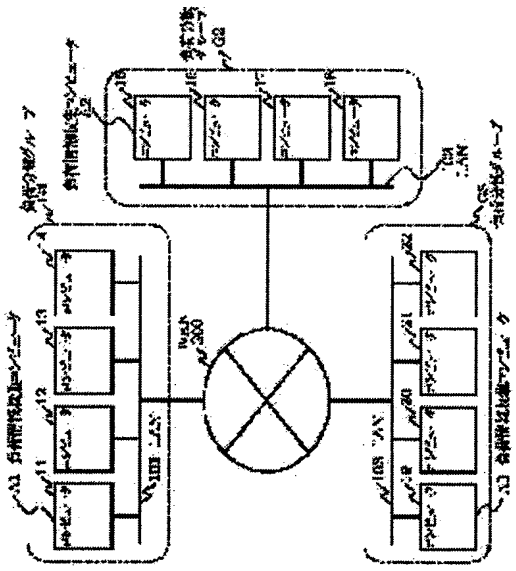
(51)Int.Cl. **G06F 13/00**
G06F 15/177

(21)Application number : **11-108156**
(22)Date of filing : **15.04.1999**

(71)Applicant : **NEC SOFTWARE KYUSHU LTD**
(72)Inventor : **TANAKA KIYOHARU**

(54) SYSTEM AND METHOD FOR LOAD DISTRIBUTION AND RECORDING MEDIUM

(57)Abstract:
PROBLEM TO BE SOLVED: To perform load distribution by grouping computers connected by a LAN(local area network), a WAN(wide area network), etc., and allowing the representative computer of the group to manage the load information of its own group and other groups.
SOLUTION: A load information collection computer A1 (computer 11) collects the loads of computers 11 to 14 in a load distribution group G1, sorts them in order of a low load, and further, prepares and stores average load information of its own group. It also acquires the average load information of respective groups from load information collection computers A2 and A3. When the computer A1 receives a processing request from a computer in its own group, it transfers the processing request to a group whose average load is lowest. In the case the group is its own group, it transfers the processing request to a computer whose load is lowest in the group.



(3)

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号
特開2000－298637
（P2000－298637A）
(43)公開日　平成12年10月24日(2000.10.24)

(51)Int.Cl.⁷		識別記号	FI		テーマコード*(参考)
Ｇ０６Ｆ	13/00	３５７	Ｇ０６Ｆ 13/00	３５７Ｚ	５Ｂ０４５
	15/177	６７２	15/177	６７２Ａ	５Ｂ０８９
		６７４		６７４Ｂ	

審査請求　有　　請求項の数７　ＯＬ　（全 11 頁）

(21)出願番号　特願平11－108156

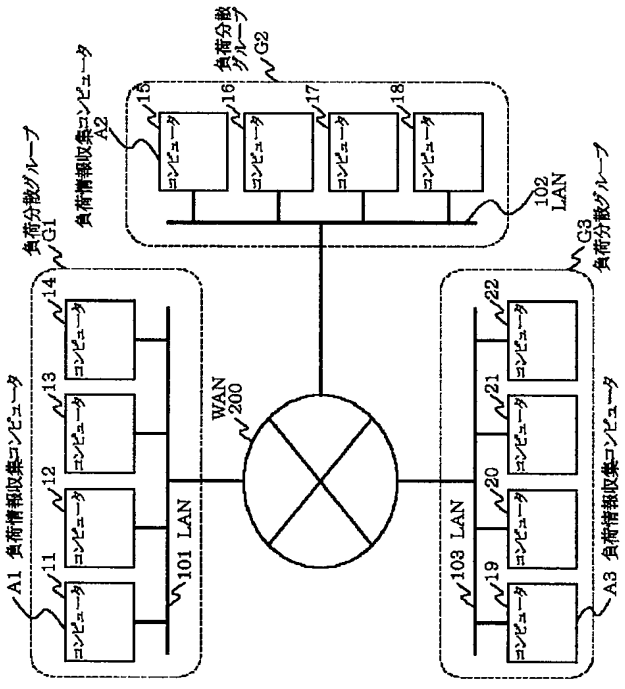
(22)出願日　平成11年４月15日(1999.4.15)

(71)出願人　000164449
九州日本電気ソフトウェア株式会社
福岡市早良区百道浜２丁目４－１　ＮＥＣ九州システムセンター
(72)発明者　田中　清春
福岡県福岡市早良区百道浜２－４－１　九州日本電気ソフトウェア株式会社内
(74)代理人　100082935
弁理士　京本　直樹　（外２名）

Ｆターム(参考)　5B045 GG04 GG09 JJ08
5B089 GA01 HA06 JA36 JB18 KA06 KB00 KB04 MA03

(54)【発明の名称】　負荷分散システム、負荷分散方法、および記録媒体

(57)【要約】
【課題】　ＬＡＮ、ＷＡＮ等で接続されたコンピュータをグループ化し、グループの代表のコンピュータが自グループ、他グループの負荷情報を管理し、負荷の分散をはかる。
【解決手段】　負荷情報収集コンピュータＡ１（コンピュータ１１）は、負荷分散グループＧ１内のコンピュータ１１～コンピュータ１４の負荷を収集し、負荷の低い順にソートし、さらに、自グループの平均負荷情報を作成し保持する。また、負荷情報収集コンピュータＡ２、負荷情報収集コンピュータＡ３から、それぞれのグループの平均負荷情報を取得する。負荷情報収集コンピュータＡ１は、自グループ内のコンピュータから処理要求を受け付けると、平均負荷の最も低いグループに対して、その処理要求を転送する。自グループであれば、自グループ内の最も負荷の低いコンピュータに処理要求を転送する。



【特許請求の範囲】

【請求項１】　１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムであって、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持する前記負荷情報収集コンピュータを有することを特徴とする負荷分散システム。

【請求項２】　自身が属する前記負荷分散グループ内の前記コンピュータの負荷がある一定値を超えた場合に、対応する前記コンピュータの情報を前記負荷情報から削除する前記負荷情報収集コンピュータを有することを特徴とする請求項１記載の負荷分散システム。

【請求項３】　前記ローカル・エリア・ネットワークに接続され、１つの前記コンピュータと同等のレベルとして扱われる下位の階層の前記ローカル・エリア・ネットワークを有することを特徴とする請求項１または２記載の負荷分散システム。

【請求項４】　自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送する前記負荷情報収集コンピュータを有することを特徴とする請求項１、２または３記載の負荷分散システム。

【請求項５】　１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させる手順を含むことを特徴とする負荷分散方法。

【請求項６】　前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むことを特徴とする請求項５記載の負荷分散方法。

【請求項７】　１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させ、かつ、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むプログラムを記録することを特徴とする記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、負荷分散システム、負荷分散方法、記録媒体に関し、特に、ネットワークで接続された複数のコンピュータに対する負荷を分散させる負荷分散システム、負荷分散方法、記録媒体に関する。

【０００２】

【従来の技術】グループ化されたコンピュータまたはプロセッサに対する負荷分散の従来技術としては、たとえば、「特開平７－９３２６５号公報」記載の技術がある。この技術は、共有メモリに接続された最下層の複数のプロセッサの負荷をこの共有メモリに書き込み、さらに、この共有メモリに接続された通信制御プロセッサがこの負荷を参照して負荷分散を実施するものである。さらに、負荷分散においては、乱数が使用される。

【０００３】
【発明が解決しようとする課題】上述した従来の技術の第１の問題点は、通信制御プロセッサが必要となることである。
【０００４】その理由は、最下層のプロセッサ同士がローカル・エリア・ネットワークで接続される構成となっていないからである。
【０００５】第２の問題点は、共有メモリが必要となることである。
【０００６】その理由は、共有メモリを介して通信制御プロセッサが負荷を参照する構成となっているからである。
【０００７】第３の問題点は、負荷分散の精度が悪いことである。
【０００８】その理由は、乱数を使用して負荷分散を行うからである。
【０００９】第４の問題点は、負荷分散のグループが任意に設定できないことである。
【００１０】その理由は、同一の階層レベル同士でしかグループを構成できないからである。
【００１１】本発明の目的は、最下層のコンピュータまたはプロセッサ同士がローカル・エリア・ネットワークで接続された構成において、通信制御プロセッサを使用せずに任意のグループに分割し、負荷分散を精度よく実施することである。
【００１２】
【課題を解決するための手段】本発明の第１の負荷分散システムは、１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムであって、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持する前記負荷情報収集コンピュータを有する。
【００１３】本発明の第２の負荷分散システムは、前記第１の負荷分散システムであって、前記自身が属する前記負荷分散グループ内の前記コンピュータの負荷がある一定値を超えた場合に、対応する前記コンピュータの情報を前記負荷情報から削除する前記負荷情報収集コンピュータを有する。
【００１４】本発明の第３の負荷分散システムは、前記第１または第２の負荷分散システムであって、前記ローカル・エリア・ネットワークに接続され、１つの前記コンピュータと同等のレベルとして扱われる下位の階層の前記ローカル・エリア・ネットワークを有する。
【００１５】本発明の第４の負荷分散システムは、前記第１、第２または第２の負荷分散システムであって、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送する前記負荷情報収集コンピュータを有する。
【００１６】本発明の第１の負荷分散方法は、１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させる手順を含む。
【００１７】本発明の第２の負荷分散方法は、前記第１の負荷分散方法であって、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含む。
【００１８】本発明の記録媒体は、１以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの１つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第１の平均負荷情報として保持し、

自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第２の平均負荷情報を収集し、前記第１の平均負荷情報、および第２の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させ、かつ、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むプログラムを記録する。

【００１９】

【発明の実施の形態】次に、本発明の第１の実施の形態について図面を参照して詳細に説明する。図１は、本発明の第１の実施の形態を示すブロック図である。図１を参照すると、本発明の第１の実施の形態は、ローカル・エリア・ネットワーク（以降、ＬＡＮと記す）であるＬＡＮ１０１と、ＬＡＮ１０１に接続されるコンピュータ１１、コンピュータ１２、コンピュータ１３、コンピュータ１４と、ＬＡＮ１０２と、ＬＡＮ１０２に接続されるコンピュータ１５、コンピュータ１６、コンピュータ１７、コンピュータ１８と、ＬＡＮ１０３と、ＬＡＮ１０３に接続されるコンピュータ１９、コンピュータ２０、コンピュータ２１、コンピュータ２２と、ＬＡＮ１０１～１０３を接続するワイド・エリア・ネットワーク（以降、ＷＡＮと記す）であるＷＡＮ２００とから構成される。

【００２０】ＬＡＮ１０１で接続されているコンピュータ１１～コンピュータ１４を負荷分散グループとして形成し、これを負荷分散グループＧ１とする。ＬＡＮ１０２で接続されているコンピュータ１５～コンピュータ１８を負荷分散グループとして形成し、これを負荷分散グループＧ２とする。ＬＡＮ１０３で接続されているコンピュータ１９～コンピュータ２２を負荷分散グループとして形成し、これを負荷分散グループＧ３とする。

【００２１】負荷分散グループＧ１内のコンピュータ１１は、自グループ内のコンピュータ１１～コンピュータ１４の負荷情報（たとえば、ＣＰＵ使用率等）および負荷分散グループＧ２、負荷分散グループＧ３の負荷情報を収集する機能と、収集した負荷情報をもとに、他グループの負荷情報を収集するコンピュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ１１を負荷分散グループＧ１の負荷情報収集コンピュータＡ１とする。

【００２２】負荷分散グループＧ２内のコンピュータ１５は自グループ内のコンピュータ１５～コンピュータ１８の負荷情報および負荷分散グループＧ１、負荷分散グループＧ３の負荷情報を収集する機能と、収集した負荷情報をもとに、他グループの負荷情報を収集するコンピュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ１５を負荷分散グループＧ２の負荷情報収集コンピュータＡ２とする。

【００２３】負荷分散グループＧ３内のコンピュータ１９は自グループ内のコンピュータ１９～コンピュータ２２の負荷情報および負荷分散グループＧ１、負荷分散グループＧ２の負荷情報を収集する機能と、収集した負荷情報をもとに他グループの負荷情報を収集するコンピュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ１９を負荷分散グループＧ３の負荷情報収集コンピュータＡ３とする。

【００２４】図１において、コンピュータ１１～コンピュータ１４は、ＬＡＮ１０１に、コンピュータ１５～コンピュータ１８は、ＬＡＮ１０２に、コンピュータ１９～コンピュータ２２は、ＬＡＮ１０３にそれぞれ接続されているが、ＬＡＮ１０１、ＬＡＮ１０２、ＬＡＮ１０３は、ＬＡＮではなくＷＡＮであってもよい。また、各グループはＷＡＮ２００を介して接続されているが、ＷＡＮ２００ではなくルータ等を介して接続されてもよい。また、各グループは一つのＬＡＮ上で構成されているが、一つのＬＡＮ上に複数のグループを構成してもよい。また、負荷情報収集コンピュータＡ１、負荷情報収集コンピュータＡ２、負荷情報収集コンピュータＡ３は、負荷分散のための処理とともに、他のコンピュータと同様に通常の処理も実行する。また、負荷情報収集コンピュータは、グループ内のどのコンピュータにも割り当て可能である。

【００２５】次に、本発明の第１の実施の形態の動作について図面を参照して説明する。図２～図６は、本発明の第１の実施の形態の動作を示すフローチャートである。図７は、負荷情報および平均負荷情報を示す説明図である。まず、図２を参照して各負荷分散グループ内での負荷情報収集の動作を説明する。

【００２６】負荷情報収集コンピュータＡ１（コンピュータ１１）は、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）からそれぞれのコンピュータの負荷情報を収集する（図２ステップＳ１）。

【００２７】自分の所属するグループの全てのコンピュータの負荷情報を収集した終えた後（図２ステップＳ２）、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする（図２ステップＳ３、図７（ａ））。ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値（たとえば、ＣＰＵ使用率８０［％］）を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する（図２ステップＳ４～Ｓ５）。

【００２８】次に、図３を参照して、負荷分散グループ

の平均負荷情報収集の動作を説明する。

【００２９】負荷情報収集コンピュータＡ１は、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～１４）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する（図３ステップＳ６～Ｓ７、図７（ｂ））。

【００３０】さらに他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２（コンピュータ１５）、負荷情報収集コンピュータＡ３（コンピュータ１９）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する（図３ステップＳ８～Ｓ１０、図７（ｃ））。

【００３１】次に、図４を参照して、各負荷分散グループへの平均負荷情報提供の動作を説明する。

【００３２】負荷情報収集コンピュータＡ１は、他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２、負荷情報収集コンピュータＡ３から平均負荷情報を要求された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータＡ２、または負荷情報収集コンピュータＡ３へ自グループ（負荷分散グループＧ１）の平均負荷情報を転送する（図４ステップＳ１１）。

【００３３】次に、図５を参照して、自グループ内のコンピュータから処理要求を受け付けた場合の負荷分散先決定動作を説明する。

【００３４】負荷情報収集コンピュータＡ１は、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内で最も負荷の低いグループから順に処理要求転送先を決定する（図５ステップＳ１２）。

【００３５】最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する（図５ステップＳ１３ＹＥＳ～Ｓ１４、たとえば、図７の例では、負荷分散グループＧ２の負荷情報収集コンピュータＡ２に対して処理要求を転送する）。

【００３６】最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する（図５ステップＳ１３ＮＯ、Ｓ１５、Ｓ１６、図７の例では、コンピュータ１２に対して処理要求を転送する）。

【００３７】次に、図６を参照して、負荷情報収集コンピュータＡ１が、他グループ（たとえば、負荷分散グループＧ２）の負荷情報収集コンピュータＡ２から処理要求を受け付けた場合の負荷情報収集コンピュータＡ２の負荷分散動作について説明する。負荷情報収集コンピュータＡ１は、ソートされた自グループ（負荷分散グループＧ１）内の負荷情報（図７（ａ））をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する（図６ステップＳ１７～Ｓ１８、図７の例では、コンピュータ１２に対して処理要求を転送する）。

【００３８】次に、本発明の第２の実施の形態について図面を参照して詳細に説明する。図８は、本発明の第２の実施の形態を示すブロック図である。図８を参照すると、本発明の第２の実施の形態は、ＷＡＮ２００を介してＬＡＮ１０２から、さらにＬＡＮ１０３が接続された構成となっている。

【００３９】この構成の場合、負荷分散グループＧ２の負荷情報収集コンピュータＡ２は、負荷分散グループＧ３を自グループに属する論理的な１台のコンピュータとして扱う。つまり、負荷情報収集コンピュータＡ２が自グループの負荷情報を収集する際、負荷分散グループＧ３の平均負荷情報を自グループ内に存在する一台のコンピュータのものとして収集する。

【００４０】また、負荷分散グループＧ１の負荷情報収集コンピュータＡ１は、他グループの平均負荷情報として負荷分散グループＧ２の平均負荷情報のみを収集する。すなわち、直接、負荷分散グループＧ３の平均負荷情報を収集しない。

【００４１】負荷分散グループＧ２の平均負荷情報は、コンピュータ１５～コンピュータ１７の負荷情報と負荷分散グループＧ３の平均負荷情報とをもとに算定される。

【００４２】また、負荷分散グループＧ３の負荷情報収集コンピュータＡ３は、自グループのコンピュータの負荷情報と、他グループ（負荷分散グループＧ１、負荷分散グループＧ２）の平均負荷情報を収集する。

【００４３】この方法により、負荷分散グループを階層化することが可能であり、負荷分散グループＧ２からさらに負荷分散グループＧ３が後で追加された場合でも、負荷分散グループＧ１はこの負荷分散グループＧ３の追加を認識する必要はない。

【００４４】次に、本発明の第３の実施の形態について図面を参照して詳細に説明する。図９は、本発明の第３の実施の形態を示すブロック図である。図９を参照すると、本発明の第３の実施の形態は、ＬＡＮ１０１とＬＡＮ１０２との間にＷＡＮ２０１、ＷＡＮ２０２が接続された構成となっている。

【００４５】このような構成で、コンピュータ１６は、コンピュータ１１～コンピュータ１４が接続されているＬＡＮ１０１とは別のＬＡＮ１０２に接続されているが、物理構成とは関係なく同一の負荷分散グループとして構成されうる。

【００４６】次に、本発明の第４の実施の形態について図面を参照して詳細に説明する。図１０は、本発明の第

４の実施の形態を示すブロック図である。図１０を参照すると、本発明の第４の実施の形態は、接続形態が本発明の第３の実施の形態と同じであるが、コンピュータ１５が二つの負荷分散グループＧ１、負荷分散グループＧ１に所属している。このように一つのコンピュータが複数の負荷分散グループに所属することも可能である。

【００４７】次に、本発明の第５の実施の形態について図面を参照して詳細に説明する。負荷情報収集コンピュータＡ１への適用例として説明する。本発明の第５の実施の形態は、負荷分散方法である。

【００４８】すなわち、本発明の第５の実施の形態は、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）からそれぞれのコンピュータの負荷情報を収集させる第１のステップ（図２ステップＳ１）と、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする第２のステップ（図２ステップＳ３、図７（ａ））と、ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する第３のステップ（図２ステップＳ４～Ｓ５）と、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～１４）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する第４のステップ（図３ステップＳ６～Ｓ７、図７（ｂ））と、他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２（コンピュータ１５）、負荷情報収集コンピュータＡ３（コンピュータ１９）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する第５のステップ（図３ステップＳ８～Ｓ１０、図７（ｃ））と、他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２、負荷情報収集コンピュータＡ３から平均負荷情報を要求された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータＡ２、または負荷情報収集コンピュータＡ３へ自グループ（負荷分散グループＧ１）の平均負荷情報を転送する第６のステップ（図４ステップＳ１１）と、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内で最も負荷の低いグループから順に処理要求転送先を決定する第７のステップ（図５ステップＳ１２）と、最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する第８のステップ（図５ステップＳ１３ＹＥＳ～Ｓ１４、たとえば、図７の例では、負荷分散グループＧ２の負荷情報収集コンピュータＡ２に対して処理要求を転送する）と、最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する第９のステップ（図５ステップＳ１３ＮＯ、Ｓ１５、Ｓ１６、図７の例では、コンピュータ１２に対して処理要求を転送する）と、ソートされた自グループ（負荷分散グループＧ１）内の負荷情報（図７（ａ））をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する第１０のステップ（図６ステップＳ１７～Ｓ１８、図７の例では、コンピュータ１２に対して処理要求を転送する）とを含む。

【００４９】次に、本発明の第６の実施の形態について図面を参照して詳細に説明する。図１１は、本発明の第６の実施の形態を示すブロック図である。図１１を参照すると、本発明の第６の実施の形態は、本発明の第５の実施の形態の負荷分散方法をコンピュータ１１～コンピュータ２２に実行させるプログラムを記録した記録媒体３００である。この記録媒体３００は磁気ディスク、半導体メモリ、その他であってよい。負荷分散方法を記録したプログラムは記録媒体３００から各コンピュータ１１～コンピュータ２２に読み込まれ、各コンピュータの負荷分散動作を制御する。各コンピュータはこの読み込まれた負荷分散方法プログラムによって本発明の第１の実施の形態で説明した処理と同一の処理を行う。

【００５０】詳細には、本発明の第６の実施の形態は、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）からそれぞれのコンピュータの負荷情報を収集させる第１のステップ（図２ステップＳ１）と、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする第２のステップ（図２ステップＳ３、図７（ａ））と、ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する第３のステップ（図２ステップＳ４～Ｓ５）と、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～１４）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する第４のステップ（図３ステップＳ６～Ｓ７、図７（ｂ））と、他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２（コンピュータ１５）、負荷情報収集コンピュータＡ３（コンピュータ１９）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する第５のステップ（図３ステップＳ８～Ｓ１０、図７（ｃ））と、他グループ（負荷分散グループＧ２、負荷分散グループＧ３）の負荷情報収集コンピュータＡ２、負荷情報収集コンピュータＡ３から平均負荷情報を要求

された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータＡ２、または負荷情報収集コンピュータＡ３へ自グループ（負荷分散グループＧ１）の平均負荷情報を転送する第６のステップ（図４ステップＳ１１）と、自分の所属するグループ（負荷分散グループＧ１）の各コンピュータ（コンピュータ１１～コンピュータ１４）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内で最も負荷の低いグループから順に処理要求転送先を決定する第７のステップ（図５ステップＳ１２）と、最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する第８のステップ（図５ステップＳ１３ＹＥＳ～Ｓ１４、たとえば、図７の例では、負荷分散グループＧ２の負荷情報収集コンピュータＡ２に対して処理要求を転送する）と、最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する第９のステップ（図５ステップＳ１３ＮＯ、Ｓ１５、Ｓ１６、図７の例では、コンピュータ１２に対して処理要求を転送する）と、ソートされた自グループ（負荷分散グループＧ１）内の負荷情報（図７（ａ））をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する第１０のステップ（図６ステップＳ１７～Ｓ１８、図７の例では、コンピュータ１２に対して処理要求を転送する）とをコンピュータに実行させるプログラムを記録した記録媒体である。

【００５１】

【発明の効果】本発明の第１の効果は、負荷情報の収集処理を行うことによるＣＰＵ負荷を複数台のコンピュータに分散させることができることである。

【００５２】その理由は、各コンピュータをグループ化し、各グループの代表のコンピュータは自グループ内の各コンピュータの負荷情報と他グループの平均負荷情報とを収集することで、負荷情報の収集処理を分散させることが可能となり、負荷情報の収集処理を行うことによるＣＰＵ負荷を複数台のコンピュータに分散させるからである。

【００５３】第２の効果は、１台のコンピュータに対する負荷情報を保持するメモリ（または記録媒体）量を押さえることができることである。

【００５４】その理由は、各コンピュータをグループ化し、各グループの代表のコンピュータが負荷情報収集処理を行うので、システム内の全コンピュータの負荷情報を１台のコンピュータで保持する必要がないからである。

【００５５】第３の効果は、各グループ間がＷＡＮ等、従量制で課金されるようなネットワークで接続されていた場合に、通信費用を押さえることができることである。

【００５６】その理由は、グループ毎に負荷情報を平均化し、他グループの代表のコンピュータはこの平均化された負荷情報のみ収集すればよいため通信データ量が少なくてすむからである。

【００５７】第４の効果は、各システムの環境設定を局所化することが可能となり、システム構築の柔軟性が高くなることである。

【００５８】その理由は、各グループの負荷情報を収集するコンピュータは、自グループ以外のコンピュータは管理しないため、別グループにコンピュータが新たに設置または撤去された場合、そのコンピュータの設置または撤去における各コンピュータへの環境設定等への影響は当該グループのみとなるからである。

【００５９】第５の効果は、コンピュータの障害を局所化することが可能となることである。

【００６０】その理由は、各グループの負荷情報を収集するコンピュータは、自グループ以外のコンピュータは管理しないため、各グループの負荷情報を収集するコンピュータ以外のコンピュータに障害が発生したとしても、その障害を検知する必要がないからである。

【００６１】また、負荷情報を収集するコンピュータが他グループの平均負荷情報の収集を停止すれば他グループから処理要求の転送が行われなくなるため、平均負荷情報の収集を停止したグループを負荷分散の対象から切り離すことが可能となり、これにより定期保守等をシステム全体に影響を及ぼすことなく行うことが可能となるからである。

【００６２】第６の効果は、構成が簡単なことである。

【００６３】その理由は、通信制御プロセッサを必要としないからである。

【図面の簡単な説明】

【図１】本発明の第１の実施の形態を示すブロック図である。

【図２】本発明の実施の形態の動作を示すフローチャートである。

【図３】本発明の実施の形態の動作を示すフローチャートである。

【図４】本発明の実施の形態の動作を示すフローチャートである。

【図５】本発明の実施の形態の動作を示すフローチャートである。

【図６】本発明の実施の形態の動作を示すフローチャートである。

【図７】負荷情報および平均負荷情報を示す説明図である。

【図８】本発明の第２の実施の形態を示すブロック図である。

【図９】本発明の第３の実施の形態を示すブロック図である。

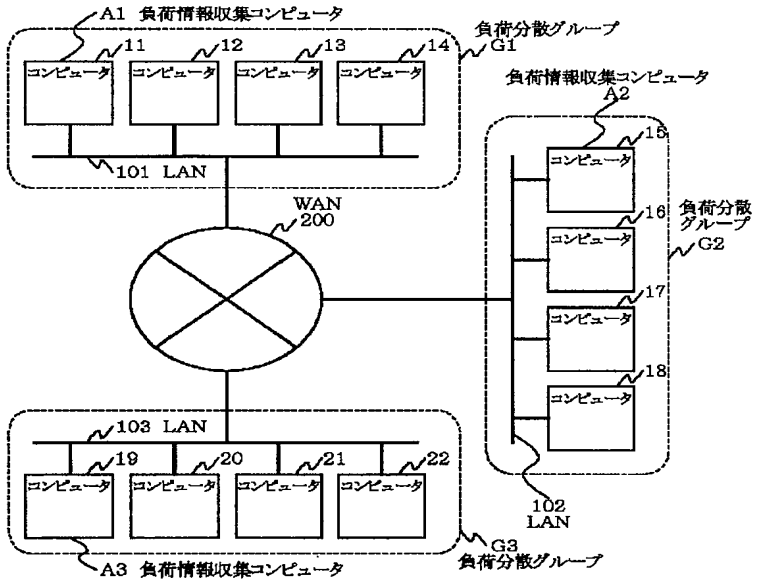
【図１０】本発明の第４の実施の形態を示すブロック図である。

【図１１】本発明の第６の実施の形態を示すブロック図である。

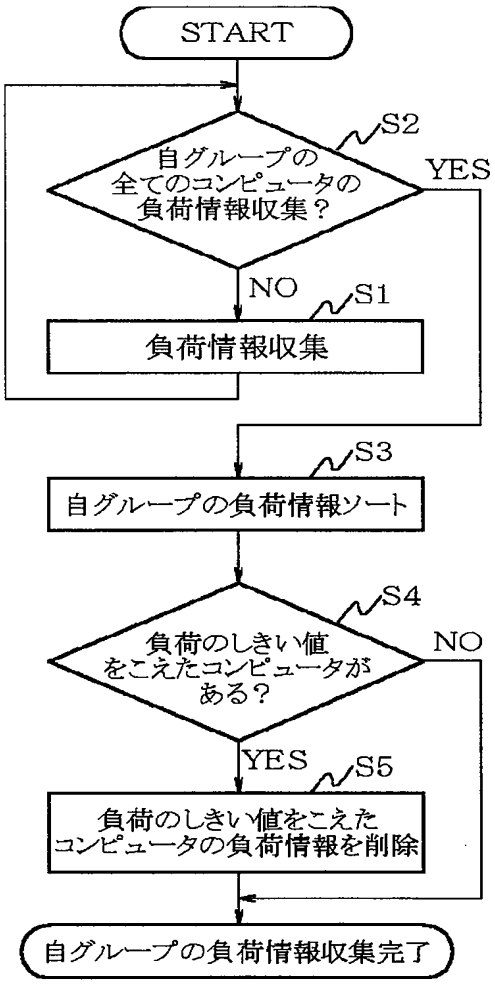
【符号の説明】

１１　コンピュータ
１２　コンピュータ
１３　コンピュータ
１４　コンピュータ
１５　コンピュータ
１６　コンピュータ
１７　コンピュータ
１８　コンピュータ
１９　コンピュータ
２０　コンピュータ
２１　コンピュータ
２２　コンピュータ
１０１　ＬＡＮ
１０２　ＬＡＮ
１０３　ＬＡＮ
２００　ＷＡＮ
２０１　ＷＡＮ
２０２　ＷＡＮ
３００　記録媒体
Ａ１　負荷情報収集コンピュータ
Ａ２　負荷情報収集コンピュータ
Ａ３　負荷情報収集コンピュータ
Ｇ１　負荷分散グループ
Ｇ２　負荷分散グループ
Ｇ３　負荷分散グループ

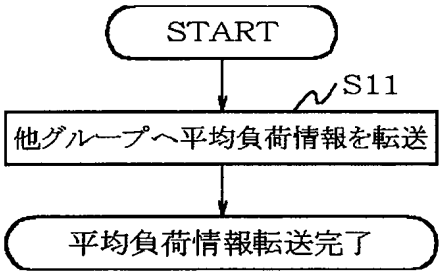
【図１】



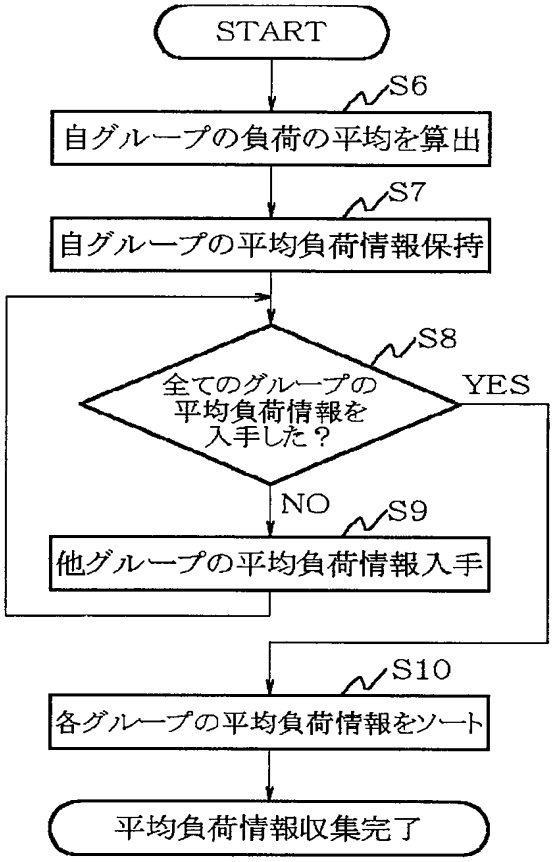
【図２】



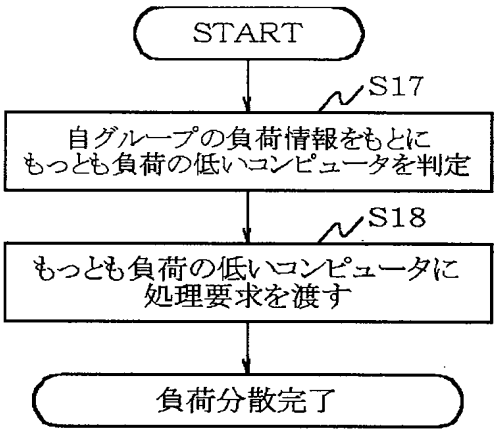
【図４】



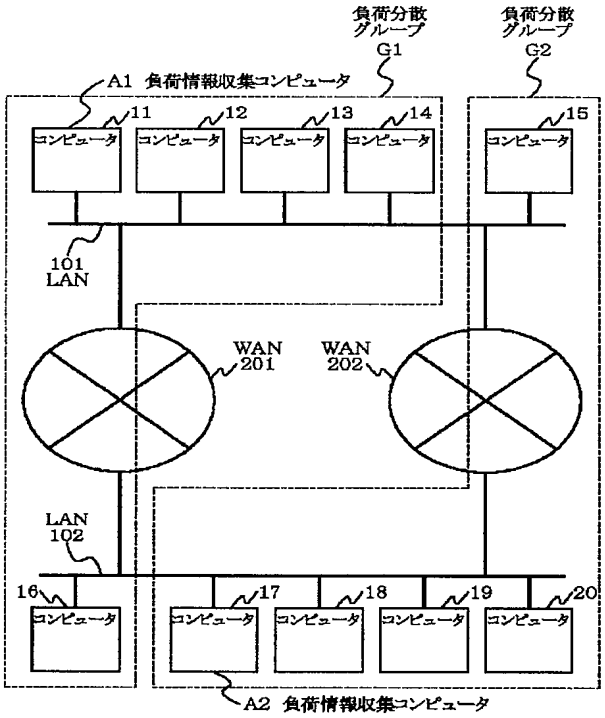
【図３】



【図６】



【図９】



【図７】

(a)

順位	CPU使用率
コンピュータ12	30
コンピュータ11	40
コンピュータ14	50
コンピュータ13	70

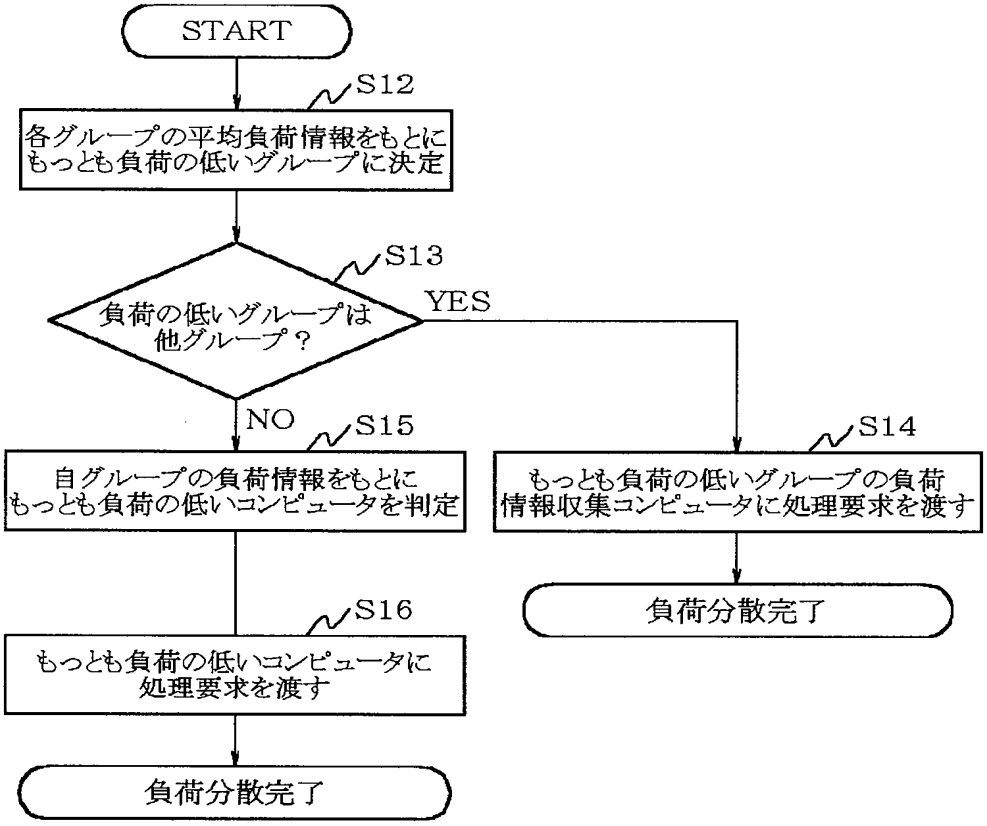
(b)

平均負荷
(30+40+50+70)／4=47.5

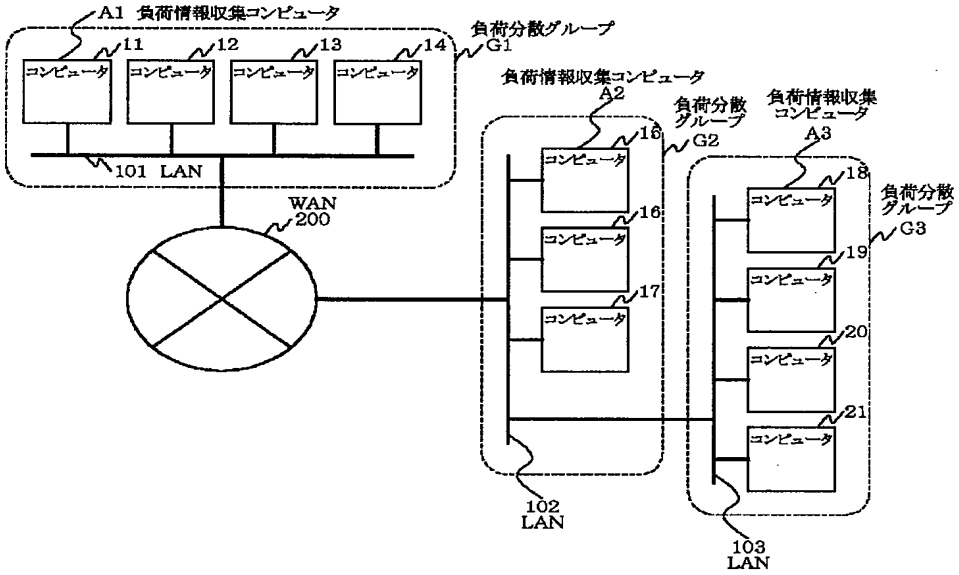
(c)

順位	平均負荷
負荷分散グループG2	30.0
負荷分散グループG1	47.5
負荷分散グループG3	50.0

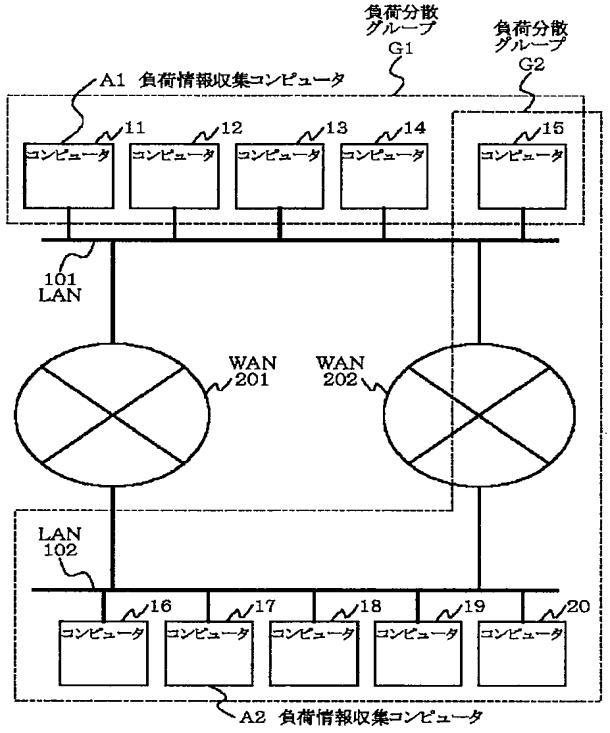
【図５】


START
S12
各グループの平均負荷情報をもとに
もっとも負荷の低いグループに決定
S13
負荷の低いグループは
他グループ？
YES
NO
S15
自グループの負荷情報をもとに
もっとも負荷の低いコンピュータを判定
S14
もっとも負荷の低いグループの負荷
情報収集コンピュータに処理要求を渡す
負荷分散完了
S16
もっとも負荷の低いコンピュータに
処理要求を渡す
負荷分散完了

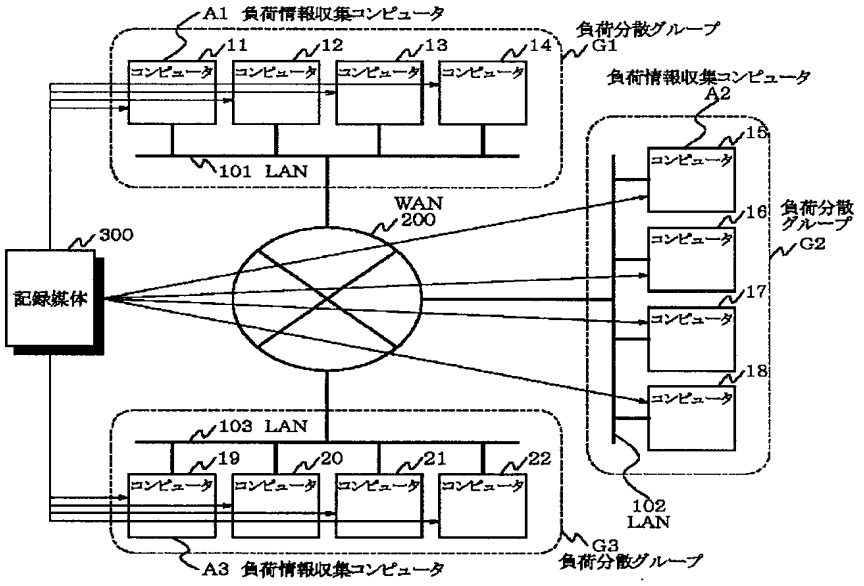
【図８】


A1 負荷情報収集コンピュータ
負荷分散グループ
G1
11
12
13
14
コンピュータ
101 LAN
WAN
200
負荷情報収集コンピュータ
A2
負荷分散
グループ
G2
16
17
102
LAN
負荷情報収集
コンピュータ
A3
18
19
20
21
負荷分散
グループ
G3
103
LAN

【図１０】


負荷分散
グループ
G1
負荷分散
グループ
G2
A1 負荷情報収集コンピュータ
11
12
13
14
15
コンピュータ
101
LAN
WAN
201
WAN
202
LAN
102
16
17
18
19
20
コンピュータ
A2 負荷情報収集コンピュータ

【図１１】


A1 負荷情報収集コンピュータ
負荷分散グループ
G1
11
12
13
14
コンピュータ
101 LAN
負荷情報収集コンピュータ
A2
15
16
17
18
コンピュータ
負荷分散
グループ
G2
WAN
200
300
記録媒体
103 LAN
19
20
21
22
102
LAN
G3
負荷分散グループ
A3 負荷情報収集コンピュータ

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2000-148783**
(43)Date of publication of application : **30.05.2000**

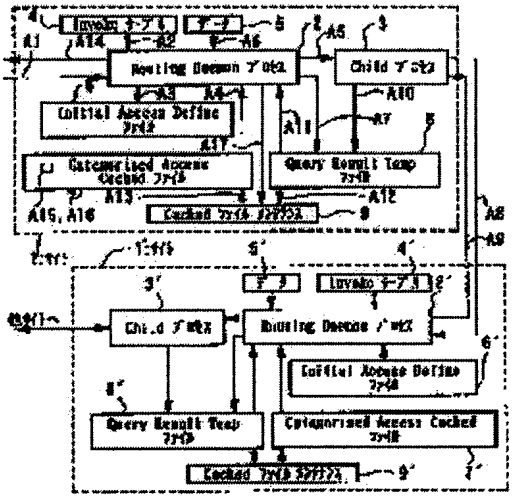
(51)Int.Cl. G06F 17/30
G06F 13/00

(21)Application number : **10-324249**
(22)Date of filing : **13.11.1998**

(71)Applicant : **NEC CORP**
(72)Inventor : **KIKUCHI SHINJI**

(54) DATA RETRIEVAL DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To widen the area of data retrieval using the internet and to make the retrieval efficient.
SOLUTION: When a site 1 receives a data retrieval request A1 from a user, a Routing Daemon process 2 actuates a Child process 3 by referring to data 5 in its device and makes another side 1' outputs a data retrieval request A8. This data retrieval request A8 includes address information A3 on a site to which the data retrieval request is always transferred and address information A4 on a site to which the most likelihood data retrieval request should be transferred from an Initial Access Define file 6. Transfer data A9 retrieved at the other side 1' are held in a Query Result Temp file 8 together with corresponding data A7 by the data 5, merged, and sent as transfer data A14 back to the user. Cached file maintenance 9 extracts the address information on the most likelihood site according to data contents transferred from the Query Result Temp file 8 and updates the Query Result Temp file 8.



(2)

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－148783
（P2000－148783A）
(43)公開日　平成12年５月30日(2000.5.30)

(51)Int.Cl.'	識別記号	ＦＩ		テーマコード'(参考)
Ｇ０６Ｆ　17/30		Ｇ０６Ｆ　15/40	３１０Ｆ	５Ｂ０７５
13/00	３５４	13/00	３５４Ｄ	５Ｂ０８９
		15/40	３１０Ｃ	

審査請求　有　　請求項の数11　ＯＬ　（全 10 頁）

(21)出願番号　特願平10－324249

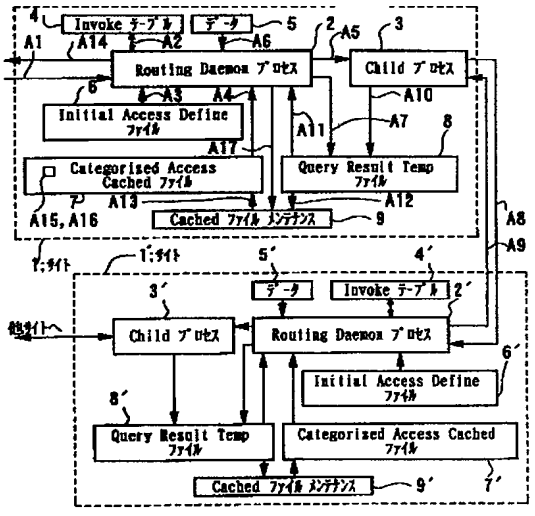
(22)出願日　平成10年11月13日(1998.11.13)

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(72)発明者　菊地　伸治
東京都港区芝五丁目７番１号　日本電気株式会社内
(74)代理人　100108578
弁理士　高橋　詔男　（外３名）

Ｆターム(参考)　5B075 KK02 KK13 KK33 PQ05 PQ20
5B089 GA11 GB09 HA10 JA12 KC15
KC23 KC39 KC44

(54)【発明の名称】　データ検索装置

(57)【要約】
【課題】　インターネットを利用したデータ検索の広域化、高効率化を図る。
【解決手段】　サイト1が利用者からデータ検索要求A1を受信すると、RoutingDaemonプロセス2は、自装置内のデータ5を参照し、Childプロセス3を起動し、他のサイト1′にもデータ検索要求A8を出力させる。このデータ検索要求A8には、InitialAccessDefineファイル6からの、常にデータ検索要求を転送する サイトのアドレス情報A3と、最も確からしいデータ検索要求を転送すべきサイトのアドレス情報A4を含む。他のサイト1′　で検索された転送データA9は、データ5による該当データA7と共にQueryResultTempファイル8に保持され、マージされ、転送データA14として利用者に返される。Cachedファイルメンテナンス9は、QueryResultTemp ファイル8から転送されて きたデータ内容に応じて最も確からしいサイトのア ドレス情報を引き出し、QueryResultTempファイル8を更新する。



【特許請求の範囲】

【請求項１】　インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、

データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置からデータを受理し、前記自装置内に保持しているデータとマージしてデータ検索要求元に転送することを特徴とするデータ検索装置。

【請求項２】　インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、

データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置から検索データを受理する常駐のRouting Daemon プロセスと、

該Routing Daemon プロセスから起動され、複数の他のデータ転送装置とデータ転送に関する通信を行なうChild プロセスとを有することを特徴とするデータ検索装置。

【請求項３】　前記各データ転送装置は、新たなデータ検索要求を受けると、前記Routing Daemon プロセスによって起動識別子が登録されるInvokeテーブルを備え、Routing Daemon プロセスは、データ検索要求を受けた場合に、該Invokeテーブルを参照することにより、重複した処理を回避することを特徴とする請求項２記載のデータ検索装置。

【請求項４】　前記各データ転送装置は、前記Routing Daemon プロセスが、常に前記データ検索要求を転送する他のデータ転送装置のアドレスを格納したInitial Access Define ファイルを備えたことを特徴とする請求項２または請求項３記載のデータ検索装置。

【請求項５】　前記各データ転送装置は、前記Routing Daemon プロセスが、前記データ検索要求のデータ内容に応じて最も確からしいデータ検索要求を転送すべきデータ転送装置のアドレスを格納したCategorized Access Cached ファイルを備えたことを特徴とする請求項２ないし請求項４のいずれかに記載のデータ検索装置。

【請求項６】　前記各データ転送装置は、前記データ検索の結果、前記自装置内で参照されたデータおよび他のデータ転送装置から転送された検索データを一時的に保持するQuery Result Temp ファイルと、該Query Result Temp ファイルから転送されてきたデータ内容に応じて最も確からしいデータ転送先のデータ転送装置のアドレスを引き出し、この結果によって前記Categorized Access Cached ファイルを更新するCached ファイル　メンテナンスとを備えたことを特徴とする請求項５記載のデータ検索装置。

【請求項７】　前記最も確からしいデータ転送装置は、まず、検索条件の指定カテゴリに対し相対網羅度が最小のものを選択し、次に、相対新規度が最小のものを第１優先条件、前記相対網羅度が最小のものを第２優先条件としてソートして決定することを特徴とする請求項５または請求項６記載のデータ検索装置。

【請求項８】　前記最も確からしいデータ転送装置の決定方法は、計算上、扱うべきデータ転送装置の数を制限するサイト上限値設定手順と、保持している検索データ件数が多い順にソートして前記制限されたデータ転送装置の数に相当する順位迄のデータ転送装置を選択する該当サイト選択手順と、検索結果の対象をソートし、同じ対象に関する記述を持つ異なるデータ転送装置の数を前記対象の各々に対して数える該当オブジェクト選択手順と、前記選択された全データ転送装置に対して、各々、データ転送装置の数を変数Ｘとし、自身を含めたＸ台のデータ転送装置に含まれる前記対象の度数に関するヒストグラムを作成するヒストグラム作成手順と、該作成されたヒストグラムを基に網羅度および新規度を計算する網羅度・新規度算出手順と、前記網羅度の小さい順にソートしてデータ転送装置を並び替え、小さい順に前記相対網羅度を付与する相対網羅度付与手順と、前記新規度の小さい順にソートしてデータ転送装置を並べ替え、小さい順に前記相対度を付与する相対新規度付与手順と、該付与された相対網羅度，相対新規度と、前記Categorized Access Cached ファイルで管理されている現在の相対網羅度,相対新規度との相対平均をとり、前記Categorized Access Cached ファイルに書き込む更新手順とを含むことを特徴とする請求項７記載のデータ検索装置。

【請求項９】　前記Child プロセスは、起動の際、データ検索要求を転送すべき他のデータ転送装置のアドレス情報および他のデータ転送装置から転送されてきたデータを一時的に保持すべき前記Query Result Temp ファイルの名称情報を前記Routing Daemon プロセスから引数として受け取ることを特徴とする請求項２ないし請求項８のいずれかに記載のデータ検索装置。

【請求項１０】　前記データ検索要求は、転送される際に、通過するデータ転送装置のName特定情報および累積Hop 数が内部にシーケンス状に記録されることを特徴とする請求項１ないし請求項９のいずれかに記載のデータ検索装置。

【請求項１１】　前記データ検索要求，Invoke テーブル，Initial AccessDefine ファイル，Categorized Access Cached ファイルおよびQuery Result Temp ファイルBNF (Backus Normal Form) で定義される記述形式で表されることを特徴とする請求項１ないし請求項１０のいずれかに記載のデータ検索装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、インターネット上

のデータ検索要求に対するデータ転送装置に関する。

【０００２】

【従来の技術】従来のこの種の転送方式を図４に示す。この方式は、米国論文（M. A. sheldon, A. Duda, R. Weiss, D. K. Gifford: Discover: A Resource Discovery System based onContent Routing, The 3rd International world-wide web Conference April10-14. 1995 (http//www.igd.de/www/www95/papers/82/html-files/discover.html) (1995).）に記されたものである。

【０００３】この従来方式は、利用者に対してサーバの役割を果たすHttpd プロセス１１，ルーティング機能を提供する複数のコンテントルータ１２等，各コンテントルータが参照し、ルーティングに関する情報を管理する複数のルーティングデータベース１３等，各コンテントルータが参照し、利用者に正しい質問入力を促進するための複数のRefinement データベース１４等，検索機能を提供する複数のサーチモジュール１６，ドキュメントデータを管理するためのシステムである複数のWAIS プロセス１７および各WAIS プロセス１７が参照する複数のWAIS データベース１８を含んでいる。

【０００４】利用者がWeb ブラウザ等を介して、Httpd プロセス１１にドキュメントの検索要求Ｂ１を発行すると、Httpd プロセス１１は、ルーティング機能を提供し、直接関連しているコンテントルータ１２に検索要求Ｂ１２を発行する。このコンテントルータ１２は、通常、Httpd プロセス１１のバックプロセスとして機能する。

【０００５】その後、コンテントルータ１２は、検索要求Ｂ１で指定され、検索要求Ｂ２に含まれる質問語彙妥当か否かを判定するため、Refinement データベース１４上の正しい質問データＢ３を参照し、妥当な表現に修正する。

【０００６】次に、コンテントルータ１２は、ルーティングデータベース１３に問い合わせ、質問語彙に対応して、該当する検索結果を応答できる、もしくはその仲立ちができる他のコンテントルータ１２′，１２″の位置情報Ｂ４を得る。この情報はルーティングデータベース１３の内部ではコンテンツラベル１５として管理されている。また他のルーティングデータベース１３′，１３″でも同様にコンテンツラベル１５′，１５″が存在している。

【０００７】次に、コンテントルータ１２は、コンテンツラベル１５から得た位置情報Ｂ４が指し示す他のコンテントルータ１２′，１２″に、検索要求Ｂ２と等価な検索要求Ｂ５，Ｂ５′を転送する。

【０００８】他のコンテントルータ１２′，１２″においても、コンテントルータ１２と全く同様の処理を行なう。その際、もしこのコンテントルータ１２′，１２″がルーティングの際に、最終位置にある場合、直接関連するサーチモジュール１６，１６′に検索要求Ｂ５，Ｂ５′と等価な検索要求Ｂ６，Ｂ６′を転送する。　サーチモジュール１６，１６′が検索要求Ｂ６，Ｂ６′を受けると、この検索要求Ｂ６，Ｂ６′と等価な検索要求Ｂ７，Ｂ７′を配下のドキュメントデータを管理するWAIS プロセス１７，１７′に発行する。それぞれのWAIS プロセス１７，１７′は検索要求Ｂ１で指定された質問語彙を含むＢ９，Ｂ９′を各WAIS データベース１８，１８′に発行し、関連するドキュメントの位置情報Ｂ１０，Ｂ１０′をWAIS プロセス１７，１７′に返す。WAIS プロセス１７，１７′は、対応するサーチモジュール１６，１６′に検索要求Ｂ７，Ｂ７′の応答として、ドキュメントの位置情報Ｂ１０，Ｂ１０′と等価な位置情報Ｂ１１，Ｂ１１′を返す。

【０００９】同様にサーチモジュール１６，１６′は対応するコンテントルータ１２′，１２″に検索要求Ｂ６，Ｂ６′の応答として、ドキュメントの位置情報Ｂ１１，Ｂ１１′と等価な位置情報Ｂ１２，Ｂ１２′を返す。その後コンテントルータ１２′，１２″は、起点となるコンテントルータ１２に検索要求Ｂ５，Ｂ５″の応答として、ドキュメントの位置情報１２，Ｂ１２′と等価な位置情報Ｂ１３，Ｂ１３′を返す。

【００１０】起点となるコンテントルータ１２は、配下のコンテントルータ１２′，１２″から検索結果であるドキュメントの位置情報Ｂ１３，Ｂ１３′を全て受けとると、それらを合成し最終的な位置情報Ｂ１４としてHttpd プロセス１１に戻す。その後、利用者が発行したドキュメントの検索要求Ｂ１に対応したドキュメントの位置情報Ｂ１５が利用者のWeb ブラウザ等上に表示されることになる。

【００１１】コンテンツラベル１５，１５′，１５″を更新する場合は、コンテントルータ１２，１２′，１２″は検索条件を限定しない検索要求Ｂ１６，Ｂ１６′を発行する。その後、前述の手順に応じて、WAIS データベース １８，１８′ 上に管理された全ドキュメントの位置情報Ｂ１７，Ｂ１７′が返されることになる。全ドキュメントの位置情報Ｂ１７，Ｂ１７′の内容はコンテンツラベル１５，１５′，１５″を更新し得るだけの内容を持ち、情報内容を持って、コンテントルータ１２，１２′，１２″はコンテンツラベル１５，１５′，１５″を更新する。

【００１２】

【発明が解決しようとする課題】上述した従来の方式２″では、図４に示した様にWAIS プロセス１７，１７′を使っているため、ドキュメント検索の際、語彙検索が中心となる。また、基本的にWAIS データベース１８，１８′で管理されるコンテンツデータを対象としており、一般的なレガシーデータを対象とはしていないという第１の問題点がある。

【００１３】また、コンテンツラベル１５，１５′，１５″を更新する手段を用意してはいるが、検索の際には

最適化した検索が行われる機構が特になく、決められたコンテントルータ１２′，１２″にのみ、常に検索要求をルーティングすることしか出来ないという第２の問題点がある。

【００１４】また、仮に、２つのルーティングデータベース１３′，１３″内の各々のコンテンツラベル１５′，１５″上に、互いに相手の位置情報が登録されている場合には、検索が２重に実施されることになり検索上のロスが発生するという第３の問題点がある。

【００１５】従って、本発明の目的は、近年、益々、重要になって来ているインターネット上のドキュメントを含むデータベースの検索を出来るだけ広域にかつ無駄なく、実施出来るデータ転送装置を提供することにある。

【００１６】また、本発明の他の目的は、利用者の欲している情報をできるだけ定量的に評価して効率を高めることができるデータ検索装置を提供することにある。

【００１７】

【課題を解決するための手段】そこで、本発明の第１のデータ検索装置は、インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置からデータを受理し、前記自装置内に保持しているデータとマージしてデータ検索要求元に転送することを特徴とする。また、本発明の第２のデータ検索装置は、インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置から検索データを受理する常駐のRouting Daemon プロセスと、該Routing Daemon プロセスから起動され、複数の他のデータ転送装置とデータ転送に関する通信を行なうChildプロセスとを有することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、新たなデータ検索要求を受けると、前記Routing Daemon プロセスによって起動識別子が登録されるInvoke テーブルを備え、Routing Daemon プロセスは、データ検索要求を受けた場合に、該Invoke テーブルを参照することにより、重複した処理を回避することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記Routing Daemon プロセスが、常に前記データ検索要求を転送する他のデータ転送装置のアドレスを格納したInitial Access Define ファイルを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記Routing Daemon プロセスが、前記データ検索要求のデータ内容に応じて最も確からしいデータ検索要求を転送すべきデータ転送装置のアドレスを格納したCategorized Access Cached ファイルを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記データ検索の結果、前記自装置内で参照されたデータおよび他のデータ転送装置から転送された検索データを一時的に保持するQuery Result Tempファイルと、該Query Result Temp ファイルから転送されてきたデータ内容に応じて最も確からしいデータ転送先のデータ転送装置のアドレスを引き出し、この結果によって前記Categorized Access Cached ファイルを更新するCached ファイル　メンテナンスとを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記最も確からしいデータ転送装置は、まず、検索条件の指定カテゴリに対し相対網羅度が最小のものを選択し、次に、相対新規度が最小のものを第１優先条件、前記相対網羅度が最小のものを第２優先条件としてソートして決定することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記最も確からしいデータ転送装置の決定方法は、計算上、扱うべきデータ転送装置の数を制限するサイト上限値設定手順と、保持している検索データ件数が多い順にソートして前記制限されたデータ転送装置の数に相当する順位迄のデータ転送装置を選択する該当サイト選択手順と、検索結果の対象をソートし、同じ対象に関する記述を持つ異なるデータ転送装置の数を前記対象の各々に対して数える該当オブジェクト選択手順と、前記選択された全データ転送装置に対して、各々、データ転送装置の数を変数Ｘとし、自身を含めたＸ台のデータ転送装置に含まれる前記対象の度数に関するヒストグラムを作成するヒストグラム作成手順と、該作成されたヒストグラムを基に網羅度および新規度を計算する網羅度・新規度算出手順と、前記網羅度の小さい順にソートしてデータ転送装置を並べ替え、小さい順に前記相対網羅度を付与する相対網羅度付与手順と、前記新規度の小さい順にソートしてデータ転送装置を並べ替え、小さい順に前記相対新規度を付与する相対新規度付与手順と、該付与された相対網羅度，相対新規度と、前記Categorized Access Cached ファイルで管理されている現在の相対網羅度,相対新規度との相対平均をとり、前記Categorized Access Cached ファイルに書き込む更新手順とを含むことを特徴とする。

【００１８】

【発明の実施の形態】次に、本発明の実施の形態について説明する。図１は、本発明のデータ内容に応じたルーティング方式を採用したデータ転送装置の実施例を示す。本実施例では、データ転送装置をサイト（Site）と称し、図１には２つのサイト１と１′が示されている。

【００１９】サイト１にはデータ検索要求を受けると、自装置内に保持しているデータを参照すると共に他サイ

ト１′へ同一のデータ検索要求を転送・発行し，その結果、他サイト１′内に保持しているデータを受理する常駐プログラムのRouting Daemon プロセス２と、Routing Daemon プロセス２から起動され、他の、サイト１′等の複数のデータ転送装置と実際のデータ転送に関する通信を行なうChild プロセス３と、データ検索要求の処理状況を管理・記録するInvoke テーブル４と、サイト内で固有に保持されているデータ５と、Routing Daemon プロセス２が常にデータ検索要求を転送・発行するサイトのアドレスを記したInitial Access Define ファイル６と、後述の該当データマイニング方式により更新され、データ検索要求を転送・発行する際に最も確からしいサイトのアドレスを得るのに使用するCategorizedAcess Cached ファイル７と、データ検索要求によるデータ５への参照結果であるデータ、並びに上述のデータ検索要求の結果、転送されて来たサイト１′内に保持されているデータ５′を一時的に保持・記録するQuery Result Temp ファイル８と、データマイニング方式により、Query Result Temp ファイル８からデータ検索要求の条件に応じて最も確からしいデータ転送先のサイトのアドレスを引き出し、その結果をもってCategorized Access Cached ファイル７を更新・維持管理するCached ファイル　メンテナンス９を含んでいる。

【００２０】Invoke テーブル４は、各サイトに１つ設定されるものであり、サイト間でやりとりされるデータ検索要求の処理状況を管理・記録するものである。RoutingDaemon プロセス２がデータ検索要求A_1を受理すると、Invoke テーブル４をアクセスし、該当する起動識別子（以下InvokeID ）を含んだ情報A_2が存在するか否かを確認する。このInvokeID を含んだ情報A_2を初めて扱う場合は、InvokeIDを含んだ情報A_2をテーブル４に書き込むことで登録する。それに対して、サイト１がサイト１′内のChild ３′から、既に同一のInvokeID に相当するデータ検索要求A_1を受けている場合は、そのInvokeID を含んだ情報は、登録済みになっているので、新たに同じデータ検索要求を受けた場合、Routing Daemon プロセス２が処理を無視する、もしくは拒否する等を行ない、２重に同一の処理が実施されることを防止する。

【００２１】Invoke テーブル４は、以下のBNF(Backus Normal Form)で定義される記述形式を持ち、同一のデータ検索要求は容易に判定出来る。

【００２２】

【数１】

```
InvokeTable ::= 〈originl_pass〉 〈invokeID〉;
〈originl_pass〉 ::= 〈identifier〉 ;
〈identifier〉 ::= 〈ip_string〉 | 〈dns_string〉 | 〈other〉 ;
〈ip_string〉 ::= {〈number〉 3} {' .' 〈number〉 3} 3;
〈dns_string〉 ::= {〈word〉' .'} {〈word〉' .'} *;
〈other〉 ::= 〈word〉 *;
〈string〉 ::= 〈character〉 | 〈character〉 〈character〉 *;
〈word〉 ::= 〈alphabetic〉 {〈alphabetic〉 *} ;
〈character〉 ::= 〈alphabetic〉 | 〈number〉 | 〈special〉 ;
〈number〉 ::=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
〈special〉 ::= 〈 | 〉 | = | + | - | * | / | & | ^ | ~ | _ | @ | $ | % | : | . | ! | ?;
〈invokeID〉 ::= 〈number〉 15 ;
```

【００２３】また、データ検索要求A_1は、以下のBNFで定義される記述形式を持ち、転送の際に通過サイトのName 特定情報、並びに累積Hop 数が内部に記憶される。尚、通過サイトのName 特定情報は、順序が把握出来るようにシーケンス状に記録される。

【００２４】

【数２】

```
QueryText ::=⟨originl_pass⟩⟨invokeID⟩⟨query⟩⟨hop_number⟩⟨pass_list⟩⟨local_pass⟩;
⟨hop_number⟩ ::= ⟨number⟩ 3;
⟨pass_list⟩ ::=' | ' {⟨remote_pass⟩' | '} *;
⟨remote_pass⟩ ::= ⟨identifier⟩ ;
⟨local_pass⟩ ::= ⟨identifier⟩ ;
⟨originl_pass⟩ ::= ⟨identifier⟩ ;
⟨identifier⟩ ::= ⟨ip_string⟩ | ⟨dns_string⟩ | ⟨other⟩ ;
⟨ip_string⟩ ::= {⟨number⟩ 3} {' .' ⟨number⟩ 3} 3;
⟨dns_string⟩ ::= {⟨word⟩' .'} {⟨word⟩' .'} *;
⟨other⟩ ::=- ⟨word⟩ *;
⟨query⟩ ::= ⟨string⟩ ;
⟨string⟩ ::= ⟨character⟩ | ⟨character⟩ ⟨character⟩ *;
⟨word⟩ ::= ⟨alphabetic⟩ {⟨alphabetic⟩ *} ;
⟨character⟩ ::= ⟨alphabetic⟩ | ⟨number⟩ ⟨special⟩ ;
⟨number⟩ ::=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
⟨special⟩ ::= ( | ) | = | + | - | * | / | & | ^ | ~ | _ | @ | $ | % | : | . | ! | ?;
⟨invokeID⟩ ::= ⟨number⟩ 15 ;
```

【００２５】Routing Daemon プロセス２は、その後、categorized Access Cached ファイル７をアクセスし、先のデータ検索要求A_1を転送・発行するべきサイトのアドレス情報A_4を得ると共に、Initial Access Define ファイル６もアクセスし、同様に先のデータ検索要求A_1を転送・発行するサイトのアドレス情報A_3を得る。Initial Access Define ファイル６には、Routing Daemon プロセス２が常にデータ検索要求A_1を転送・発行するサイトのアドレスが記されているのに対して、Categorized Access Cached ファイル７には、データ検索要求A_1で指定された検索条件としてのカテゴリに応じて、データ検索要求を転送・発行する際に最も確からしいサイトのアドレスが記されている。

【００２６】その際の最も確からしいサイトの決定手順は、図２に記されたようになる。ステップＳ１として、まず、後述の相対網羅度A_{15}が最小のものを、利用者が検索条件として指定したカテゴリに対し、最も有効と思われるサイトとして選択する。相対網羅度A_{15}が最小ということは、もっとも多くの該当するデータを保持しているサイトを意味する。

【００２７】次に、ステップＳ２では、利用者が検索条件として、指定したカテゴリに対し、以下の優先条件でソートし、複数のサイトを規定数分、選択する。

第一優先条件）相対新規度Ａ１６が最小のもの

第二優先条件）相対網羅度Ａ１５が最小のもの

この条件設定の理由は、可能な限り、どのサイトも扱っていないものを検索対象にするためである。

【００２８】前述の様にInitial Access Define ファイル６には、サイト１内のChild プロセス３が、常にデータ検索要求A_1を転送・発行するべきアドレス情報Ａ３が指定されているが、このInitial Access Define ファイル６は、以下の様なBNF で表される記述形式を持つ。

【００２９】

【数３】

```
InitialAccessDefinefile::= {⟨indentifier⟩' | '} *;
⟨identifier⟩ ::= ⟨ip_string⟩ | ⟨dns_string⟩ | ⟨other⟩ ;
⟨ip_string⟩ ::= {⟨number⟩ 3} {' .' ⟨number⟩ 3} 3;
⟨dns_string⟩ ::= {⟨word⟩' .'} {⟨word⟩' .'} *;
⟨other⟩ ::=- ⟨word⟩ *;
```

【００３０】次に、Routing Daemon プロセス２は、サイトデータ検索要求Ａ１を転送・発行するべきサイトのアドレス情報Ａ３，Ａ４にChild プロセス３を該当数分だけ立ち上げる。起動の際、各々のChild サイトプロセス３はデータ検索要求A_1を転送・発行するべきアドレス情報Ａ３，Ａ４，並びにデータ検索要求Ａ１の結果、転送されて来た内に保持されているデータ５′を、一時的に保持・記録するQuery Result Temp ファイル８の名称情報を引数Ａ５として受け取る。任意のChild プロセス３がサイト１′に該データ検索要求Ａ１と等価なデータ検索要求Ａ８を転送・発行する際は、サイト１′内のRouting Daemon プロセス２′が受理する。その際、データ検索要求Ａ８にはサイト１のアドレス情報が付加され、データ検索要求Ａ８内部のHop数が１つ追加される。

【００３１】次に、Routing Daemon プロセス２は、自サイト内で管理しているデータ５からデータ検索要求Ａ１に応じたデータのサブセットＡ６を受けとり、一時的に保持・記録する為、のQuery Result Temp ファイル８に該当データＡ７として書き込む。Query Result Temp ファイル８は、以下のBNFで表される記述形式を持ち、プロセス３ごとに１つ作成される。

【００３２】

【数４】

```
QueryResultTempFile::= 〈header〉 {〈URLdescription〉 *} ;
〈header〉 ::= 〈querytext〉 〈amount〉 ;
〈amount〉 ::= 〈number〉 6;
〈URLdescription〉 ::= 〈URL〉 〈object_identifier〉 〈description〉 ;
〈URL〉 ::=' http://' {〈ip_string〉 | 〈dns_string〉}' /' 〈string〉 ;
〈object_identifier〉 ::= 〈string〉 ;
〈description〉 ::= 〈string〉 ;
```

【００３３】サイト１′内でもサイト１と同様に、Routin Daemon プロセス２′が同じ処理を行ない、サイト１′内で管理しているデータ５′からデータ検索要求Ａ８に応じたデータのサブセットＡ６を引き出し、転送データＡ９としてサイト内のChild プロセス３に戻す。Child プロセス３は転送データＡ９を受理すると、引数Ａ５で指定されたQuery Result Temp ファイル８に該当データＡ１０として書き込む。この処理は、起動されたChild プロセス３数分だけ実施される。

【００３４】次に、Routing Daemon プロセス２は全てのChild プロセス３が該当データＡ１０を書き込んだことを確認すると、全てのQuery Result Temp ファイル８をマージし、そこから該当データＡ１１を読み出す。その後、Routing Daemon プロセス２は当該データＡ１１をデータ検索要求Ａ１に応じた転送データＡ１４として、データ検索要求元に転送する。以上で、一連のデータ検索要求Ａ１は完了する。Routing Daemon プロセス２は、不要となったQuery Result Temp ファイル８を消去する際に、Cached ファイル　メンテナンス９へ起動要求１７を送付する。Cached ファイル　メンテナンス９は起動すると、データ検索結果を格納したQuery Result Temp ファイル８から、Categorized Access Cached ファイル７上の相対網羅度Ａ１５，相対新規度Ａ１６を更新するためのパラメータデータＡ１２を引き出し、これを基に更新元データＡ１３を作成し、Categorized Access Cached ファイル７上の現相対網羅度Ａ１５、現相対新規度Ａ１６を更新する。前述の様にCategorized Access Cached ファイル７とは、データ検索要求Ａ１，Ａ８を転送した結果、得られる一時的なデータ検索格納先であるQuery Result Temp ファイル８から、事前に定義した当該データのマイニング方式に応じて、各サイトの評価を行ない、そのアドレス情報を管理するものである。Categorized Access Cached ファイル７により、Routing Daemon プロセス２は、データ検索要求Ａ１，Ａ８を転送・発行する際に、検索条件として指定されるカテゴリに対し最も有効と思われるサイトのみに転送・発行先を絞り込むことが出来るので、全サイトへ単純にブロードキャスト転送を行なうよりは、遥かに効果的なルーティング転送・発行処理を実現することが出来る。Categorized Access Cached ファイル７は以下のBNF で表される記述形式を持つ。

【００３５】

【数５】

```
CategorizedAssessCachedfile::= {〈registry description〉' |'} *;
〈registry description〉 ::= 〈category identifier〉 〈identifier〉 〈metrix description〉 ;
〈category identifier〉 ::= 〈string〉 ;
〈identifier〉 ::= 〈ip_string〉 | 〈dns_string〉 | 〈other〉 ;
〈ip_string〉 ::= {〈number〉 3} {' .' 〈number〉 3} 3;
〈dns_string〉 ::= {〈word〉' .'} {〈word〉' .' } *;
〈other〉 ::= 〈word〉 *;
〈metrix description〉 ::= 〈relative covering degree〉 〈relative strange degree〉 〈counter〉 ;
〈relative covering degree〉 ::= 〈number〉 6;
〈relative strange degree〉 ::= 〈number〉 6;
〈counter〉 ::= 〈number〉 6;
```

【００３６】Categorized Access Cached ファイル７上で管理される相対網羅度Ａ１５，相対新規度Ａ１６は、以下に記されるものである。

【００３７】１）相対網羅度Ａ１５

Cached ファイル　メンテナンス９は当該データのマイニング方式としてサイト１′を始めとする関連する複数のサイトがデータ検索要求Ａ１，Ａ８に相当する任意データ検索要求を受理した結果として戻す転送データＡ９，Ａ１４に対し、各サイトがどの程度、検索条件である指定カテゴリに該当するものを含んでいるかを計算し、更新元データＡ１３を求める。その後、その計算値である更新元データＡ１３で、Categorized Access Cached ファイル７上で管理されるサイト，カテゴリ毎に記録している相対網羅度Ａ１５の値を更新する。相対網羅度Ａ１５の評価に関しては後述する。

【００３８】２）相対新規度Ａ１６

Cached ファイル　メンテナンス９は、前述の当該データのマイニング方式として、サイト１′を始めとする関連する複数のサイトがデータ検索要求Ａ１，Ａ８に相当する任意データ検索要求を受理した結果として戻す転送

データＡ９．Ａ１４に対し、各サイトがどの程度、検索条件である指定カテゴリに該当するもので独自かつ固有なものを含んでいるかを計算し、更新元データＡ１３を求める。その後その計算値である更新元データＡ１３で、Categorized Access Cached ファイル７上で管理される毎に記録している相対新規度Ａ１６の値を更新する。相対新規度Ａ１６の評価式に関しては後述する。

【００３９】次に、当該データのマイニング方式について説明する。図３は、該当データのマイニング方式のフローチャートである。ステップＳ１では、初期処理として、計算上、扱うべきサイト数を制限する。これは転送データＡ９を戻すサイトは基本的に不特定多数であることから配慮されている。これを「サイト上限値ステップ」と呼ぶ。

【００４０】ステップＳ２はChild プロセス３毎に作成されるQuery Result Temp ファイル８から、<header>記述を集める処理である。<header>記述内部には、データ件数を意味する<amount>が記されている。

【００４１】ステップＳ３では，<amount>記述中の値で大きい順にソートして<header>記述を並べ替え、ステップＳ１で制限されたサイト数に相当する順位のもの迄の複数サイトを処理の対象として選択する。これを「該当選択ステップ」と呼ぶ。

【００４２】ステップＳ４では、選択した複数サイトに対応するQuery Result Temp ファイル８中の<URLdescription>記述を全て取り出し、１つの<URLdescription>記述を１レコードと見なして、単純に連結する。なお、<URLdescription>記述の内部には、検索結果の対象を意味する<object identifier>記述が記されており、各々が１つの検索対象と見なされる。

【００４３】ステップＳ５では，<object identifier>記述でソートし、同じ<object identifier>記述を持つ異なるサイトの数を求める。以上を異なる<object identifier>記述毎全てにわたり実施する（ステップＳ６）。これを「該当object 選択ステップ」と呼ぶ。

【００４４】ステップＳ７では、ステップＳ３で選択した全サイトに対して（ステップＳ１６）、各々、サイト数を変数Ｘとし、自身を含めたＸ台のサイトに含まれる<object identifier>記述の度数ｍ（Ｘ）に関するヒストグラムを作成する。このステップＳ６では、ステップＳ３で選択した全サイトにわたり実施する。これを「ヒストグラム作成ステップ」と呼ぶ。

【００４５】ステップＳ８では、Ｓ７で求めたヒストグラムを基に綱羅度及び新規度を以下の様に計算する。これを「網羅度・新規度算出ステップ」と呼ぶ。ステップＳ８も、ステップ７と同様に、ステップＳ３で選択した全サイトに対して行なう（ステップＳ１７）。

【００４６】

【数６】

$$\text{網羅度} = \sum_{x} \{m(X) * X\}$$

$$\text{新規度} = \sum_{x} \{m(X)/X\}$$

【００４７】ステップＳ９では、ステップＳ８で求めた網羅度の小さい順に選択したサイトをソートして、並べ替える。その後、小さい順に相対網羅度を１位，２位，３位と付与する。これを「相対網羅度付与ステップ」と呼ぶ。

【００４８】ステップＳ１０では、ステップＳ８で求めた新規度の小さい順に選択したサイトをソートして、並べ替える。その後、小さい順に相対新規度を１位，２位，３位と付与する。これを「相対新規度付与ステップ」と呼ぶ。

【００４９】ステップＳ１１では、ステップＳ９並びにステップＳ１０で求めた相対綱羅度Ａ１５、相対新規度Ａ１６を更新値Ａ１３とする。その後、Categorized AccessCached ファイル７上でサイト，カテゴリ毎に管理されている現在の相対綱羅度Ａ１５、現在の相対新規度Ａ１６を読み出す。これを「既存データ読みだしステップ」と呼ぶ。

【００５０】ステップＳ１３では、以下の処理を行なう。ステップＳ１１で現在の相対綱羅度Ａ１５，現在の相対新規度Ａ１６が読み出せる場合、該当する各々と更新値Ａ１３との相加平均を取り、新たな相対綱羅度Ａ１５、相対新規度Ａ１６を求める。その際、相加平均の対象数を表す<Counter>記述値に１を加える。その後、Categorized Access Cached ファイル７に新たに算出した相対綱羅度Ａ１５、並びに新たに算出した相対新規度Ａ１６並びに<Counter>記述を書き込み、更新する。

【００５１】ステップＳ１１で現在の相対綱羅度Ａ１５、現在の相対新規度Ａ１６が読み出せない場合は、ステップＳ９で求めた相対綱羅度Ａ１５、ステップＳ９で求めた相対新規度Ａ１６、並びに該<Counter>記述を１として、Categorized Access Cached ファイル７に書き込み、更新する（ステップＳ１４）。これを「更新ステップ」と呼ぶ。

【００５２】ステップＳ１１並びにＳ１３，Ｓ１４は、ステップＳ３で選択した全サイト数回、処理を行う（ステップＳ１５）。その後、規定回数分の処理を終えたならば、該当データのマイニング方式全体の処理を終了する。

【００５３】

【発明の効果】本発明によれば、従来のように、語彙検索のみを対象とはしていないため、数値表現を含んだデータの検索が可能であり、一般的なレガシーデータをもその対象とすることが可能であるという第１の効果を有する。

【００５４】また、本発明では、データのマイニング方式を採用しているので、動的にルーティングを実施出来、その結果、新たなサイトが登録され、それが該当デ

ータのマイニング方式の評価に対して妥当なデータを戻す場合は、新たなルーティング対象として、これを取り込むことが可能であるという第２の効果を有する。

【００５５】さらに、本発明は、データ検索要求の起動状況を管理・記録するInvoke テーブルが実装されているので、検索が２重になることはないという第３の効果も有する。

【図面の簡単な説明】

【図１】　本発明のデータ検索装置の一実施例の構成図。

【図２】　本発明におけるルーティング先を決定する際の手順を示すフローチャート。

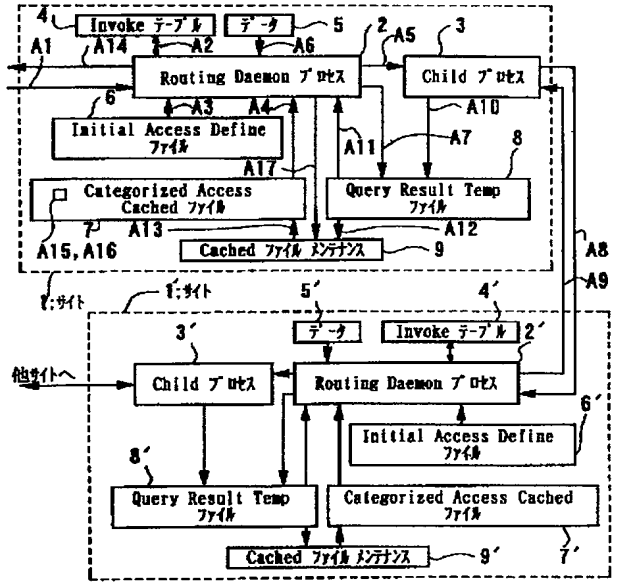
【図３】　本発明における該当データのマイニング方式の手順を示すフローチャート。

【図４】　従来方式の概要図。

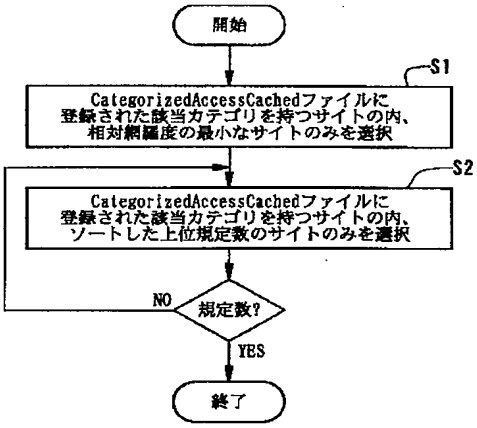
【符号の説明】

１　データ転送装置（サイト）
２　Routing Daemon プロセス
３　Child プロセス
４　Invoke テーブル
５　データ
６　Initial Access Define ファイル
７　Categorized Access Cached ファイル
８　Query Result Temp ファイル
９　Cached ファイル メンテナンス
１１　Httpd プロセス
１２　コンテントルータ
１３　ルーティングデータベース
１４　Refinement データベース
１５　コンテンツラベル
１６　サーチモジュール
１７　WAIS プロセス
１８　WAIS データベース

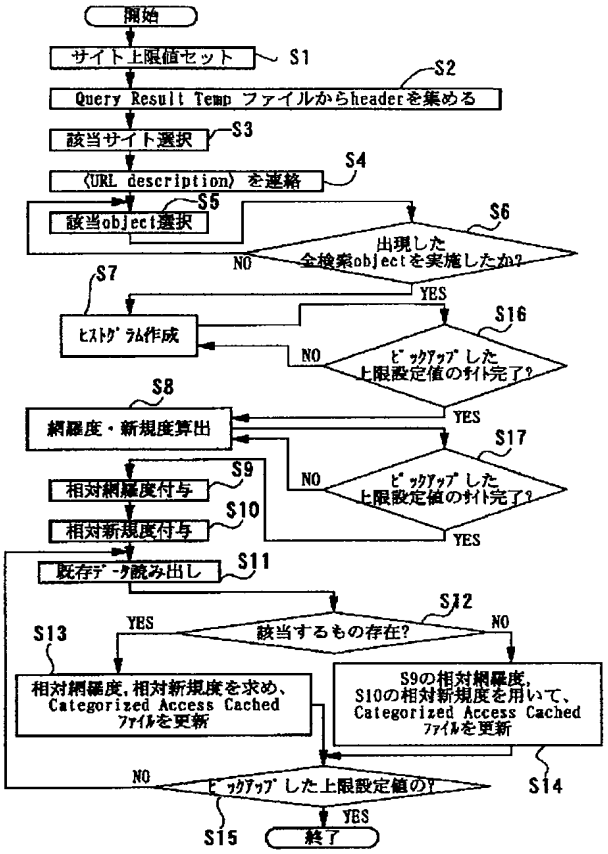
【図１】



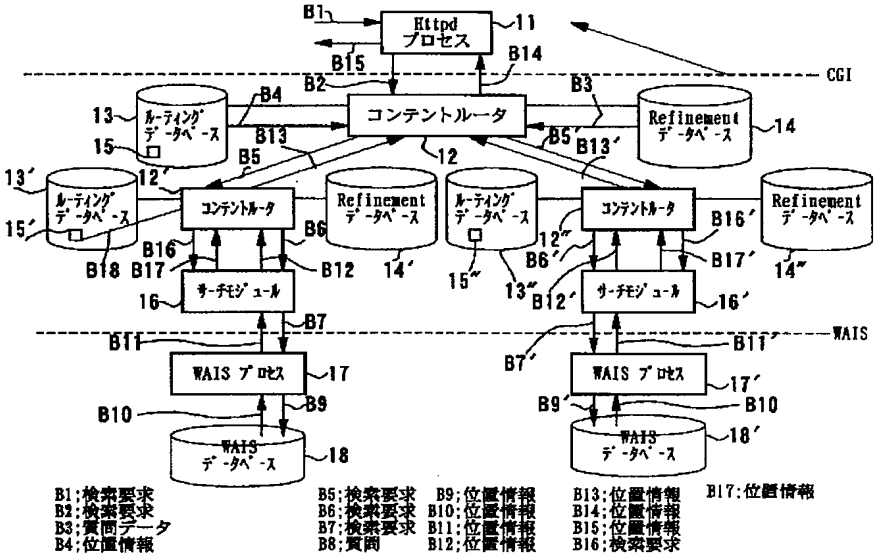
【図２】



【図３】


開始
サイト上限値セット
S1
S2
Query Result Temp ファイルからheaderを集める
該当サイト選択
S3
〈URL description〉を連絡
S4
該当object選択
S5
出現した全検索objectを実施したか?
S6
NO
YES
S7
ヒストグラム作成
S16
ピックアップした上限設定値のサイト完了?
NO
YES
S8
網羅度・新規度算出
S17
ピックアップした上限設定値のサイト完了?
NO
YES
S9
相対網羅度付与
S10
相対新規度付与
既存データ読み出し
S11
S12
該当するもの存在?
YES
NO
S13
相対網羅度,相対新規度を求め、Categorized Access Cached ファイルを更新
S9の相対網羅度,S10の相対新規度を用いて、Categorized Access Cached ファイルを更新
S14
ピックアップした上限設定値の?
NO
YES
S15
終了

【図４】


B1
Httpd プロセス
11
B15
B14
B2
B3
CGI
B4
13
ルーティング データベース
コンテントルータ
Refinement データベース
14
15
B13
B5
B13′
12
13′
12′
B5
ルーティング データベース
コンテントルータ
Refinement データベース
ルーティング データベース
コンテントルータ
Refinement データベース
15′
B16
B6
B16′
12″
B18
B17
B12
14′
B6′
B17′
14″
15″
13″
B12′
16
サーチモジュール
サーチモジュール
16′
B7
B11
WAIS
B7′
B11′
WAIS プロセス
17
WAIS プロセス
17′
B10
B9
B9′
B10
WAIS データベース
18
WAIS データベース
18′
B1;検索要求
B2;検索要求
B3;質問データ
B4;位置情報
B5;検索要求
B6;検索要求
B7;検索要求
B8;質問
B9;位置情報
B10;位置情報
B11;位置情報
B12;位置情報
B13;位置情報
B14;位置情報
B15;位置情報
B16;検索要求
B17;位置情報

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2001-160024**
(43)Date of publication of application : **12.06.2001**

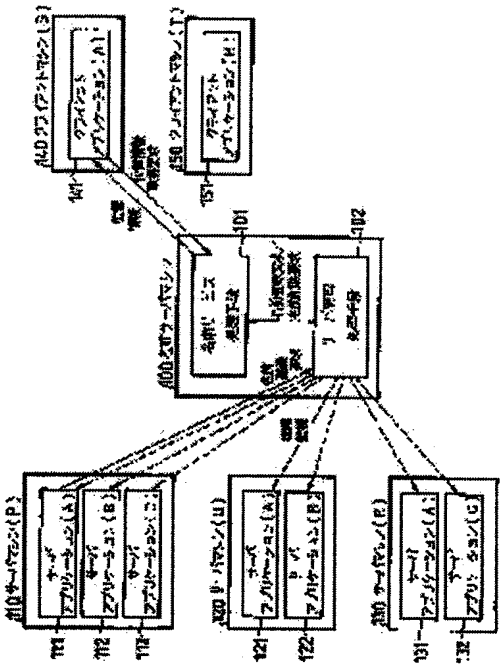
(51)Int.Cl. **G06F 13/00**

(21)Application number : **11-342927**
(22)Date of filing : **02.12.1999**

(71)Applicant : **NEC CORP**
(72)Inventor : **HIRAMOTO KAZUYOSHI**

(54) SYSTEM FOR SELECTING MANAGEMENT OF SERVER APPLICATION

(57)Abstract:
PROBLEM TO BE SOLVED: To allow a client application to correctly select and connect only connectable server applications.
SOLUTION: A name server processing means 101 returns position information in name service registration information, in response to a request for obtaining position information from a client application, registers name service registration information at accepting of a request for name registration, and deletes the name service registration information at accepting of a request for name deletion. A server management processing means 102 registers server management information at accepting of a request for name registration from the server application and transfers the request for name registration to the name server processing means 101, and normally monitors the state of the registered server application, and issues a request for name deletion of the server application to a server processing means 101 at the time of detecting the failure of the server application.



(4)

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2001－160024
（P2001－160024A）
(43)公開日　平成13年６月12日(2001.6.12)

(51)Int.Cl.[7]	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０６Ｆ　13/00	３５７	Ｇ０６Ｆ　13/00	３５７Ｚ　５Ｂ０８９

審査請求　有　　請求項の数８　ＯＬ　（全 11 頁）

(21)出願番号　特願平11－342927

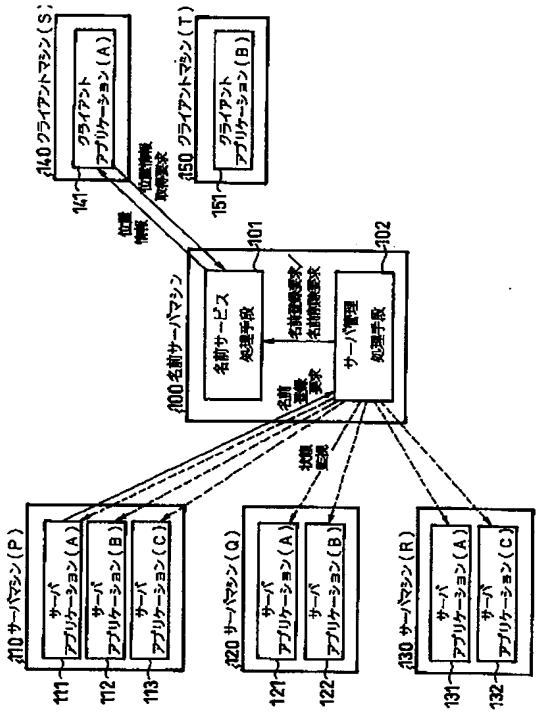
(22)出願日　平成11年12月２日(1999.12.2)

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(72)発明者　平本　和好
東京都港区芝五丁目７番１号　日本電気株式会社内
(74)代理人　100088890
弁理士　河原　純一

Ｆターム(参考)　5B089 GA11 GA21 GB02 JA11 JA35
JB17 KA12 KB04 KB06 KC28
KC51 KH03 ME06

(54)【発明の名称】　サーバアプリケーションの管理選択方式

(57)【要約】
【課題】接続可能なサーバアプリケーションのみをクライアントアプリケーションが正しく選択して接続できるようにする。
【解決手段】名前サービス処理手段１０１は、クライアントアプリケーションからの位置情報取得要求に対して登録している名前サービス登録情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス登録情報を登録し、名前削除要求を受け付けて名前サービス登録情報を削除する。サーバ管理処理手段１０２は、サーバアプリケーションからの名前登録要求を受け付けてサーバ管理情報を登録するとともに名前サービス処理手段１０１に名前登録要求を転送し、登録されたサーバアプリケーションに対しては定常的に状態監視を行い、サーバアプリケーションの障害を検出した場合に名前サービス処理手段１０１に対して当該サーバアプリケーションの名前削除要求を発行する。



【特許請求の範囲】

【請求項１】サーバアプリケーションを提供する任意の数のサーバマシン，サーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン，およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおいて、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段と、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段とを備えることを特徴とするサーバアプリケーションの管理選択方式。

【請求項２】前記サーバ管理処理手段が、サーバアプリケーションからの名前登録要求を受け付けて監視対象情報にリンクして名前サービス登録情報を登録し該名前登録要求を前記名前サービス処理手段に転送する名前登録部と、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録部を介して前記名前サービス処理手段に発行するサーバ監視部とを含んで構成される請求項１記載のサーバアプリケーションの管理選択方式。

【請求項３】前記名前サービス登録情報が、サーバアプリケーションの名前と、該サーバアプリケーションの位置情報とから構成される請求項１または２記載のサーバアプリケーションの管理選択方式。

【請求項４】前記名前サービス管理情報が、サーバアプリケーションの名前と、該名前にリンクされたサーバアプリケーションの位置情報とから構成される請求項１または２記載のサーバアプリケーションの管理選択方式。

【請求項５】前記監視対象情報が、監視対象の単位をサーバアプリケーションとした場合にサーバマシンの名前とサーバアプリケーションの名前との組である請求項１または２記載のサーバアプリケーションの管理選択方式。

【請求項６】前記監視対象情報が、監視対象の単位をサーバマシンとした場合にサーバマシンの名前である請求項１または２記載のサーバアプリケーションの管理選択方式。

【請求項７】コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段，およびサーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段として機能させるためのプログラムを記録した記録媒体。

【請求項８】コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段，サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録手段，および各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録手段を介して前記名前サービス処理手段に発行するサーバ監視手段として機能させるためのプログラムを記録した記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はサーバアプリケーションの管理選択方式に関し、特にサーバアプリケーションを搭載する任意の数のサーバマシン，サーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン，およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおけるサーバアプリケーションの管理選択方式に関する。

【０００２】

【従来の技術】名前サーバマシンを利用するネットワークシステムでは、サーバアプリケーションが名前および位置情報を名前サーバマシンに登録し、クライアントアプリケーションは名前をキーに名前サーバマシンにサーバアプリケーションの位置情報を取得要求し、返却されたサーバアプリケーションの位置情報を用いてサーバアプリケーションに接続して運用を行う。

【０００３】従来の名前サーバマシンを利用するネットワークシステムでは、サーバアプリケーションが名前および位置情報の登録後に障害等を起こし接続不可となった場合でも、名前サーバマシンにサーバアプリケーションの名前および位置情報が残り、クライアントアプリケーションからのサーバアプリケーションの位置情報取得要求時に接続不可のサーバアプリケーションの位置情報

を返却してしまうという不具合があった。

【０００４】そこで、特開平６－６８００７号公報に開示された「クライアント・サーバ連結装置」では、サーバアプリケーションが位置情報を名前サーバマシンに定期的に再登録することで、名前サーバマシンにおけるサーバアプリケーションの位置情報を最適化している。また、サーバアプリケーションの障害等をクライアントアプリケーションで認識した場合は、クライアントアプリケーションがサーバアプリケーションの位置情報を名前サーバマシンから削除することによっても名前サーバマシンにおけるサーバアプリケーションの位置情報を更新している。

【０００５】

【発明が解決しようとする課題】しかし、上述した従来の技術には、次のような問題点があった。

【０００６】第１の問題点は、障害等を起こして接続不可になったサーバアプリケーションへの接続を回避するためには、名前サーバマシンを始めとする各処理手段に機能を追加する等、システム全体に変更を加えなくては実現できないことである。その理由は、システムの各部で問題回避の処理を行っているためである。

【０００７】第２の問題点は、状態維持のためのコストがかかることである。その理由は、すべてのサーバアプリケーションが本来一度で済むべき名前登録処理を定期的に繰り返すため、それによるシステム負荷およびネットワーク負荷がサーバアプリケーション数および接続時間に比例してかかってしまうためである。

【０００８】第３の問題点は、サーバアプリケーションの名前が不当に削除される可能性があることである。その理由は、クライアントアプリケーションでサーバアプリケーションの不正状態を検出した場合に、そのままサーバアプリケーションの名前を削除してしまうからである。このため、クライアントアプリケーションの状態の如何によっては、正常に動作しているサーバアプリケーションの名前をも削除しかねなかった。

【０００９】本発明の目的は、接続可能なサーバアプリケーションのみをクライアントアプリケーションが正しく選択して接続できるようにしたサーバアプリケーションの管理選択方式を提供することにある。

【００１０】

【課題を解決するための手段】本発明のサーバアプリケーションの管理選択方式は、サーバアプリケーションを提供する任意の数のサーバマシン，これらサーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン，およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおいて、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段と、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段とを備えることを特徴とする。

【００１１】さらに、本発明のサーバアプリケーションの管理選択方式は、前記サーバ管理処理手段が、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録部と、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録部を介して前記名前サービス処理手段に発行するサーバ監視部とを含んで構成されることを特徴とする。

【００１２】一方、本発明の記録媒体は、コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段，およびサーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段として機能させるためのプログラムを記録する。

【００１３】また、本発明の記録媒体は、コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段，サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録手段，および各監視対象を定常的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録手段を介して前記名前サービス処理手段に発行するサーバ監視手段として機能させるためのプログラムを記録する。

【００１４】本発明のサーバアプリケーションの管理選択方式では、名前サーバマシンは、通常の名前サービス

を行う名前サービス処理手段の他に、サーバ管理処理手段を有し、名前サービス処理手段に登録された各サーバマシン上の各サーバアプリケーションは、サーバ管理処理手段によって定常的に状態監視がなされる。これにより、あるサーバアプリケーションが障害等により接続不可となった場合には、サーバ管理処理手段が当該サーバアプリケーションの名前削除要求を名前サービス処理手段に速やかに発行するため、クライアントアプリケーションからのサーバアプリケーションの位置情報取得要求に対して、名前サーバマシンが接続不可となったサーバアプリケーションの位置情報をクライアントアプリケーションに返却することがなくなる。クライアントアプリケーションにおいては、名前サービス処理手段からサーバアプリケーションの位置情報取得要求は、従来の名前サーバマシンへの位置情報取得要求と同様に発行することが可能である。このようにして、従来の名前サービスを利用したネットワークシステムの形態に極力手を加えずに、接続可能なサーバアプリケーションの位置情報のみを名前サービス処理手段に残すことを可能にする。また、監視対象を調整することにより、監視のための通信負荷を低減したりシステム全体の性能にも配慮することが可能である。

【００１５】

【発明の実施の形態】以下、本発明の実施の形態について図面を参照して詳細に説明する。

【００１６】図１は、本発明の第１の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムの要部を示すブロック図である。このネットワークシステムは、サーバアプリケーション（Ａ）１１１，サーバアプリケーション（Ｂ）１１２およびサーバアプリケーション（Ｃ）１１３を搭載するサーバマシン（Ｐ）１１０と、サーバアプリケーション（Ａ）１２１およびサーバアプリケーション（Ｂ）１２２を搭載するサーバマシン（Ｑ）１２０と、サーバアプリケーション（Ａ）１３１およびサーバアプリケーション（Ｃ）１３２を搭載するサーバマシン（Ｒ）１３０と、サーバアプリケーション（Ａ）１１１，１２１および１３１に接続して運用するクライアントアプリケーション（Ａ）１４１を搭載するクライアントマシン（Ｓ）１４０と、サーバアプリケーション（Ｂ）１１２および１２２に接続して運用するクライアントアプリケーション（Ｂ）１５１を搭載するクライアントマシン（Ｔ）１５０と、各サーバアプリケーションの名前サービス管理情報を登録し、クライアントアプリケーション（Ａ）１４１およびクライアントアプリケーション（Ｂ）１５１からの位置情報取得要求に応じて接続可能な１つのサーバアプリケーションの位置情報を返却する名前サービスを提供する名前サーバマシン１００とから、その主要部が構成されている。なお、括弧書きのアルファベットは名前を表し、同じ名前のサーバアプリケーションは同一のアプリケーションであり、いずれも同じ名前のクライアントアプリケーションから接続されて運用可能なものである。

【００１７】名前サーバマシン１００は、名前サービス処理手段１０１と、サーバ管理処理手段１０２とを含んで構成されている。

【００１８】名前サービス処理手段１０１は、サーバ管理処理手段１０２からの名前登録要求を受け付け、名前サービス管理情報を登録する。また、名前サービス処理手段１０１は、サーバ管理処理手段１０２からの名前削除要求を受け付け、名前サービス管理情報を削除する。さらに、名前サービス処理手段１０１は、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の名前にリンクされた位置情報のうちの１つを返却する。

【００１９】サーバ管理処理手段１０２は、各サーバアプリケーションからの名前登録要求を受け付け、監視対象情報にリンクさせてサーバアプリケーションの名前および位置情報からなる名前サービス登録情報を登録するとともに、名前登録要求を名前サービス処理手段１０１に転送する。また、サーバ管理処理手段１０２は、監視対象情報に登録された監視対象（サーバマシン，サーバアプリケーション等）に対して定常的に状態監視を行う。さらに、サーバ管理処理手段１０２は、状態監視により監視対象の障害等を検出した場合、当該監視対象のサーバアプリケーションの名前削除要求を名前サービス処理手段１０１に速やかに発行する。

【００２０】図２を参照すると、サーバ管理処理手段１０２は、各サーバアプリケーションより名前登録要求を受け付けて名前登録要求を名前サービス処理手段１０１に転送する名前登録部１０２１と、監視対象情報を基に各監視対象を定期的に状態監視するサーバ監視部１０２２と、サーバ管理情報を格納するサーバ管理情報格納部１０２３とを備えている。

【００２１】名前登録部１０２１は、あらかじめ監視対象情報３１０（図３参照）をサーバ管理情報格納部１０２３に登録する。また、名前登録部１０２１は、サーバアプリケーションからの名前登録要求を受け付けると（図２①参照）、名前登録要求から名前サービス登録情報３２０（図３参照）を作成し監視対象情報３１０にリンクさせてサーバ管理情報格納部１０２３に登録する（図２②参照）。すなわち、名前登録部１０２１は、名前サービス登録情報３２０を、予め規定しておいた監視対象の単位（サーバマシン，サーバアプリケーション等）に従って監視対象情報３１０にリンクさせてサーバ管理情報としてサーバ管理情報格納部１０２３に登録する。この後、名前登録部１０２１は、サーバアプリケーションからの名前登録要求を名前サービス処理手段１０１に転送する（図２③参照）。

【００２２】サーバ監視部１０２２は、サーバ管理情報

格納部１０２３のサーバ管理情報を検索して監視対象情報３１０を取得し（図２④参照）、監視対象情報３１０に基づいて各監視対象を状態監視する（図２⑤参照）。状態監視時に監視対象の障害等を検出した場合は、サーバ監視部１０２２は、サーバアプリケーションに代わって名前削除要求を名前登録部１０２１を介して名前サービス処理手段１０１に発行する（図２⑥参照）。

【００２３】図３を参照すると、サーバ管理情報格納部１０２３のサーバ管理情報は、監視対象情報３１０と、監視対象情報３１０にリンクされた１つ以上の名前サービス登録情報３２０とから構成されている。

【００２４】監視対象情報３１０は、サーバ管理処理手段１０２が監視対象を状態監視するために必要な情報を格納する。格納される情報は、監視対象の種類によって変化する。監視対象の単位を最小単位であるサーバアプリケーションとした場合は、サーバアプリケーションの固有情報となるサーバマシンの名前とサーバアプリケーションの名前との組が監視対象情報となる。また、監視対象の単位をサーバマシンとした場合は、サーバマシンの名前が監視対象情報となる。その他、サーバマシングループ，サーバアプリケーショングループ等を監視対象の単位とすることが可能である。

【００２５】名前サービス登録情報３２０は、名前登録要求があったサーバアプリケーションの名前と、その位置情報との組で構成される。なお、位置情報の後の括弧内のアルファベットは、サーバマシンの名前およびサーバアプリケーションの名前の組からなる位置情報の値を示す（以下同様）。名前サービス登録情報３２０は、新規にサーバアプリケーションから名前登録要求を受け付けた場合、必ず作成される情報であり、サーバ管理処理手段１０２が各サーバアプリケーションより受け付けた名前登録要求の内容の記録であるとともに、名前サービス処理手段１０１に登録した内容の記録である。

【００２６】図２を参照すると、名前サービス処理手段１０１は、名前サービス管理情報格納部１０１１を備えている。

【００２７】図４を参照すると、名前サービス管理情報格納部１０１１は、サーバ管理情報格納部１０２３に登録されたサーバアプリケーションの名前４１０と、その名前４１０を持つ１つ以上のサーバアプリケーションの位置情報４２０とを管理する。１つの名前４１０に対して位置情報４２０を複数持つ場合、クライアントアプリケーションから位置情報取得要求された名前４１０に対して管理している位置情報４２０のうちの１つを巡回方式で選択して返却する。

【００２８】図５を参照すると、名前登録部１０２１の名前登録時の処理は、サーバ管理情報更新ステップ２１と、名前登録要求転送ステップ２２と、名前登録要求受け付け判定ステップ２３と、状態監視開始ステップ２４と、サーバ管理情報巻き戻しステップ２５とからなる。

【００２９】図６を参照すると、サーバ監視部１０２２の障害検出時の処理は、接続不可監視対象検出ステップ３１と、名前サービス登録情報検索ステップ３２と、名前削除要求発行ステップ３３と、監視対象除外ステップ３４とからなる。

【００３０】次に、このように構成された第１の実施の形態に係るサーバアプリケーションの管理選択方式の動作について詳細に説明する。

【００３１】まず、名前登録時の処理について説明する。

【００３２】サーバ管理処理手段１０２の名前登録部１０２１は、各サーバアプリケーションからの名前登録要求を待ち受けている。

【００３３】名前登録部１０２は、あるサーバアプリケーションから名前登録要求を受け付けると（図２①参照）、名前登録要求中のサーバアプリケーションの名前および位置情報を名前サービス登録情報３２０として該当する監視対象情報３１０にリンクさせてサーバ管理情報格納部１０２３に登録する（図２②参照）（ステップ２１）。

【００３４】次に、名前登録部１０２は、受け付けた名前登録要求を名前サービス処理手段１０１に転送する（図２③参照）（ステップ２２）。

【００３５】名前サービス処理手段１０１は、名前登録要求を受け付けると、名前登録要求中のサーバアプリケーションの名前が名前サービス管理情報格納部１０１１にすでに登録されていれば、名前登録要求中のサーバアプリケーションの位置情報を該当する名前にリンクさせて名前サービス管理情報格納部１０１１に登録する。名前登録要求中のサーバアプリケーションの名前が名前サービス管理情報格納部１０１１に登録されていなければ、名前サービス処理手段１０１は、名前登録要求中のサーバアプリケーションの名前および位置情報をリンクさせて名前サービス管理情報として名前サービス管理情報格納部１０１１に登録する。この後、名前サービス処理手段１０１は、名前登録要求に基づく名前登録が正常に行われたか否かを名前登録結果として名前登録部１０２１に返却する。

【００３６】名前登録部１０２１は、名前サービス処理手段１０１から返却された名前登録結果に基づいて名前登録要求が受け付けられたかどうかを判断し（ステップ２３）、受け付けられたのであれば、要求元のサーバアプリケーションに名前の登録完了を返却し、以後、監視対象の状態監視を開始する（ステップ２４）。

【００３７】一方、名前登録要求が受け付けられなかったのであれば、名前登録部１０２１は、要求元のサーバアプリケーションに名前の登録失敗を通知し、サーバ管理情報格納部１０２３のサーバ管理情報を巻き戻す（ステップ２５）。

【００３８】次に、障害等による接続不可検出時の処理

について説明する。

【００３９】サーバ管理処理手段１０２のサーバ監視部１０２２は、サーバ管理情報格納部１０２３のサーバ管理情報を検索して監視対象情報３１０を取得し（図２④参照）、監視対象情報３１０に基づいて監視対象を定常的に状態監視する（図２⑤参照）。

【００４０】サーバ監視部１０２２は、ある監視対象が障害等により接続不可になったことを検出すると（ステップ３１）、サーバ管理情報格納部１０２３から当該監視対象の監視対象情報３１０にリンクされた名前サービス登録情報３２０を検索し（ステップ３２）、この名前サービス登録情報３２０中の名前の名前削除要求を名前登録部１０２１を介して名前サービス処理手段１０１に発行する（図２⑥参照）（ステップ３３）。

【００４１】名前サービス処理手段１０１は、名前削除要求を受け付けると、名前削除要求中のサーバアプリケーションの名前が名前サービス管理情報格納部１０１１にすでに登録されていれば、名前登録要求中のサーバアプリケーションの名前４１０および該名前４１０にリンクされた全ての位置情報４２０を名前サービス管理情報格納部１０１１から削除する。この後、名前サービス処理手段１０１は、名前削除要求に基づく名前削除が正常に行われたか否かを名前削除結果として名前登録部１０２１に返却する。

【００４２】名前登録部１０２１は、名前サービス処理手段１０１から返却された名前削除結果を受け付けると、サーバ監視部１０２２に転送する。

【００４３】サーバ監視部１０２２は、名前削除結果を受け付けると、サーバ管理情報格納部１０２３のサーバ管理情報中の該当する監視対象情報３１０およびそれにリンクされた名前サービス登録情報３２０を削除し、以降の監視対象から除外する（ステップ３４）。

【００４４】次に、具体例を用いてより詳細に説明する。

【００４５】（１）まず、監視対象がサーバアプリケーションである場合を例にとって説明する。

【００４６】図７に示すように、サーバアプリケーション（Ａ）１１１，１２１および１３１がサーバマシン（Ｐ）１１０，サーバマシン（Ｑ）１２０およびサーバマシン（Ｒ）１３０上でそれぞれ稼動し、サーバアプリケーション（Ｂ）１１２および１２２がサーバマシン（Ｐ）１１０およびサーバマシン（Ｑ）１２０上でそれぞれ稼動し、サーバアプリケーション（Ｃ）１１３および１３２がサーバマシン（Ｐ）１１０およびサーバマシン（Ｒ）１３０上でそれぞれ稼動しているものとする。また、クライアントマシン（Ｓ）１４０上で稼動するクライアントアプリケーション（Ａ）１４１は、サーバアプリケーション（Ａ）に接続して動作するものとする。

【００４７】図７の場合、監視対象の単位をサーバアプリケーションとしているが、これは、図８に示すように、監視対象情報をサーバアプリケーション単位に構築することによって実現される。すなわち、監視対象情報３１０は、サーバマシンの名前およびサーバアプリケーションの名前の組で構成され、新規に名前登録要求があったサーバアプリケーション毎に１つ作成される。同時に登録内容である名前サービス登録情報３２０が１対１で対応付けられて管理される。

【００４８】まず、クライアントアプリケーション（Ａ）１４１が、名前サーバマシン１００上の名前サービス処理手段１０１に対してサーバアプリケーション（Ａ）の位置情報取得要求を発行したものとする。

【００４９】名前サービス処理手段１０１は、通常状態では、クライアントアプリケーション（Ａ）１４１からのサーバアプリケーション（Ａ）の位置情報取得要求に対して、名前サービス管理情報格納部１０１１に格納されている名前４１０にリンクされている位置情報４２０のうちの１つを返却する。具体的には、図４に例示するように、名前サービス管理情報格納部１０１１には、サーバアプリケーション（Ａ）の名前（Ａ）にリンクしてサーバアプリケーション（Ａ）１１１の位置情報（ＰＡ），サーバアプリケーション（Ａ）１２１の位置情報（ＱＡ）およびサーバアプリケーション（Ａ）１３１の位置情報（ＲＡ）の３つの位置情報が登録されているので、名前サービス処理手段１０１は、３つの位置情報（ＰＡ），（ＱＡ）および（ＲＡ）のうちから１つを選択してクライアントアプリケーション（Ａ）１４１に返却する。

【００５０】ここで、サーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１が障害等により接続不可となったものとする。

【００５１】各サーバアプリケーションは、サーバ管理処理手段１０２のサーバ監視部１０２２によって監視されている。

【００５２】サーバ監視部１０２２は、サーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１が障害等により接続不可になったことを検出すると（ステップ３１）、サーバマシン（Ｑ）１２０の名前（Ｑ）およびサーバアプリケーション（Ａ）１２１の名前（Ａ）をキーとしてサーバ管理情報格納部１０２３に格納されているサーバ管理情報から名前サービス登録情報３２０を検索し（ステップ３２）、検索された名前サービス登録情報３２０に基づいてサーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１の名前サービス管理情報を削除する名前削除要求を作成し名前登録部１０２１を介して名前サービス処理手段１０１に発行する（ステップ３３）。

【００５３】名前サービス処理手段１０１は、名前削除要求を受け付けると、名前削除要求の対象である名前（Ａ）および該名前（Ａ）にリンクされた位置情報（ＱＡ）からなる名前サービス管理情報を名前サービス管理

情報格納部１０１１から削除する。すなわち、サーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１の名前サービス管理情報を削除する。

【００５４】名前サービス処理手段１０１から名前削除結果が返却されると、サーバ監視部１０２２は、サーバ管理情報格納部１０２３からサーバマシン（Ｑ）１２０の監視対象情報３１０にリンクされたサーバアプリケーション（Ａ）１２１の名前（Ａ）および位置情報（ＱＡ）からなる名前サービス登録情報を削除する（ステップ３４）。

【００５５】よって、この後、名前サービス処理手段１０１は、クライアントアプリケーション（Ａ）１４１からサーバアプリケーション（Ａ）の位置情報取得要求があっても、サーバマシン（Ｑ）１２０上のサーバアプリケーション（Ａ）１２１の位置情報（ＱＡ）を選択して返却することはなく、サーバマシン（Ｐ）１１０上のサーバアプリケーション（Ａ）１１１の位置情報（ＰＡ）かサーバマシン（Ｒ）１３０上のサーバアプリケーション（Ａ）１３１の位置情報（ＲＡ）かを選択して返却する。

【００５６】（２）　次に、監視対象の単位がサーバマシンである場合を例にとって説明する。

【００５７】図９に示すように、サーバアプリケーション（Ａ）１１１，１２１および１３１がサーバマシン（Ｐ）１１０，サーバマシン（Ｑ）１２０およびサーバマシン（Ｒ）１３０上でそれぞれ稼動し、サーバアプリケーション（Ｂ）１１２および１２２がサーバマシン（Ｐ）１１０およびサーバマシン（Ｑ）１２０上でそれぞれ稼動し、サーバアプリケーション（Ｃ）１１３および１３２がサーバマシン（Ｐ）１１０およびサーバマシン（Ｒ）１３０上でそれぞれ稼動しているものとする。

【００５８】図９の場合、監視対象の単位をサーバマシンとしているが、これは、図１０に示すように、監視対象情報３１０をサーバマシンを単位に構築することによって実現する。すなわち、監視対象情報３１０は、サーバマシンの名前からなり、新規に名前登録要求があったサーバマシン毎に作成される。また、このときに、１つ目の名前サービス登録情報３２０が監視対象情報３１０にリンクされる。以降、すでに名前登録要求の実績があるサーバマシンからの名前登録要求の受付時には、前に作成された監視対象情報３１０に新しい名前サービス登録情報３２０がリンクされる。

【００５９】いま、クライアントアプリケーション（Ａ）１４１が、サーバアプリケーション（Ａ）の位置情報取得要求を名前サービス処理手段１０１に発行したとする。

【００６０】名前サービス処理手段１０１は、通常状態では、クライアントアプリケーション（Ａ）１４１からのサーバアプリケーション（Ａ）の位置情報取得要求に対して、名前サービス管理情報格納部１０１１に名前サービス管理情報が格納されているサーバマシン（Ｐ）１１０の位置情報（ＰＡ），サーバマシン（Ｑ）１２０の位置情報（ＱＡ），およびサーバマシン（Ｒ）１３０の位置情報（ＲＡ）のうちから１つを選択してクライアントアプリケーション（Ａ）１４１に返却する。

【００６１】サーバ管理処理手段１０２のサーバ監視部１０２２は、サーバマシン（Ｐ）１１０，サーバマシン（Ｑ）１２０およびサーバマシン（Ｒ）１３０を状態監視している。本例では、監視対象の単位がサーバアプリケーションでなくサーバマシンであるため、監視対象が広がることにより、状態監視に必要な通信コストが抑えられる。

【００６２】ここで、サーバマシン（Ｑ）１２０が障害等によりダウンして接続不可になったものとする。

【００６３】サーバ監視部１０２２は、サーバマシン（Ｑ）１２０が障害等によりダウンして接続不可となったことを検出すると（ステップ３１）、サーバ管理情報格納部１０２３のサーバ管理情報からサーバマシン（Ｑ）１２０の監視対象情報３１０にリンクされたすべての名前サービス登録情報３２０を検索し（ステップ３２）、サーバマシン（Ｑ）１２０のすべてのサーバアプリケーション（Ａ）１２１およびサーバアプリケーション（Ｂ）１２２の名前サービス管理情報を削除する名前削除要求を名前登録部１０２１を介して名前サービス処理手段１０１に発行する（ステップ３３）。

【００６４】名前サービス処理手段１０１は、名前削除要求を受け付けると、名前サービス管理情報格納部１０１１の名前サービス管理情報からサーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１およびサーバアプリケーション（Ｂ）１２２の名前サービス管理情報を削除する。

【００６５】名前サービス処理手段１０１から名前削除結果が返却されると、サーバ監視部１０２２は、サーバ管理情報格納部１０２３からサーバ管理情報を削除する（ステップ３４）。

【００６６】よって、この後、クライアントアプリケーション（Ａ）１４０から、サーバアプリケーション（Ａ）に対する位置情報取得要求があっても、サーバマシン（Ｑ）１２０のサーバアプリケーション（Ａ）１２１を選択することはなく、サーバマシン（Ｐ）１１０のサーバアプリケーション（Ａ）１１１かサーバマシン（Ｒ）１３０のサーバアプリケーション（Ａ）１３１かを選択して返却することになる。

【００６７】次に、本発明の第２の実施の形態について図面を参照して説明する。

【００６８】図１１を参照すると、本発明の第２の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムは、図１に示した第１の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムに対して、名前

サーバマシン１００が名前サーバプログラムを記録した記録媒体２００を備える点だけが異なっている。したがって、対応する手段等には同一の符号を付してそれらの詳しい説明を省略する。なお、記録媒体２００は、磁気ディスク，半導体メモリ，その他の記録媒体であってよい。

【００６９】このように構成された第２の実施の形態に係るサーバアプリケーションの管理選択方式では、名前サーバプログラムが記録媒体２００からコンピュータでなる名前サーバマシン１００に読み込まれ、名前サービス処理手段１０１およびサーバ管理処理手段１０２として名前サーバマシン１００の動作を制御する。名前サーバマシン１００の詳しい動作は、第１の実施の形態に係るサーバアプリケーションの管理選択方式における場合と全く同様になるので、その詳しい説明を割愛する。

【００７０】ところで、上記各実施の形態では、監視対象の単位をサーバアプリケーションまたはサーバマシンとした場合について説明したが、監視対象の単位をサーバアプリケーショングループとしても、サーバマシングループとしても、本発明を同様に適用することができる。例えば、監視対象の単位をサーバアプリケーショングループとした場合、他のサーバアプリケーションに大きな影響を与えるサーバアプリケーションの障害を検出したときに、これに依存する全てのサーバアプリケーションの名前を削除することが可能となる。また、監視対象の単位をサーバマシングループとした場合、サーバマシンの障害を検出した際に、当該サーバマシングループから登録された全てのサーバアプリケーションの名前を削除することも可能である。このように、監視対象の単位を調整したり、それらを組み合わせて使うことで、より柔軟で効率的な監視処理を行うことが可能となる。

【００７１】

【発明の効果】第１の効果は、障害等により接続不可となったサーバアプリケーションにクライアントアプリケーションが接続にいくことを未然に回避できることである。その理由は、サーバ管理処理手段により接続不可のサーバアプリケーションの名前は名前サーバマシンから削除されており、クライアントアプリケーションは接続不可のサーバアプリケーションの位置情報を取得することがないためである。

【００７２】第２の効果は、既存の名前サービスを利用するネットワークシステムにそのまま適用できることにある。その理由は、クライアントアプリケーションに特別な機能を追加することも、クライアントマシンに特別な処理手段を追加することも必要がないため、既存のクライアント環境に手を加えることなくそのまま使い続けることが可能である。また、名前サービスも登録要求元が各サーバアプリケーションからサーバ管理処理手段に変わるだけなので、既存の名前サービスにも影響を与えない。サーバアプリケーションにおいても、位置情報の登録処理に関しては、登録要求先が名前サービス処理手段からサーバ管理処理手段にかわる以外に影響はない。

【００７３】第３の効果は、監視対象を切り替えることで効率の良い監視ができることである。その理由は、監視対象をサーバアプリケーション毎，サーバアプリケーショングループ毎，サーバマシン毎，サーバマシングループ毎というように切り替えることにより、１つ障害で効率よく名前サービスへの更新を行うことができるためである。

【図面の簡単な説明】

【図１】本発明の第１の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムの要部を示すブロック図である。

【図２】図１中の名前サーバマシンのより詳細な構成を示すブロック図である。

【図３】図２中のサーバ管理情報格納部の内容を例示する図である。

【図４】図２中の名前サービス管理情報格納部の内容を例示する図である。

【図５】図２中の名前登録部の名前登録時の処理を示すフローチャートである。

【図６】図２中のサーバ監視部の障害検出時の処理を示すフローチャートである。

【図７】図１に示した第１の実施の形態に係るサーバアプリケーションの管理選択方式の運用例を示す図である。

【図８】図７に示した運用例でのサーバ管理情報格納部の内容例を示す図である。

【図９】図１に示した第１の実施の形態に係るサーバアプリケーションの管理選択方式の他の運用例を示す図である。

【図１０】図９に示した運用例でのサーバ管理情報格納部の内容例を示す図である。

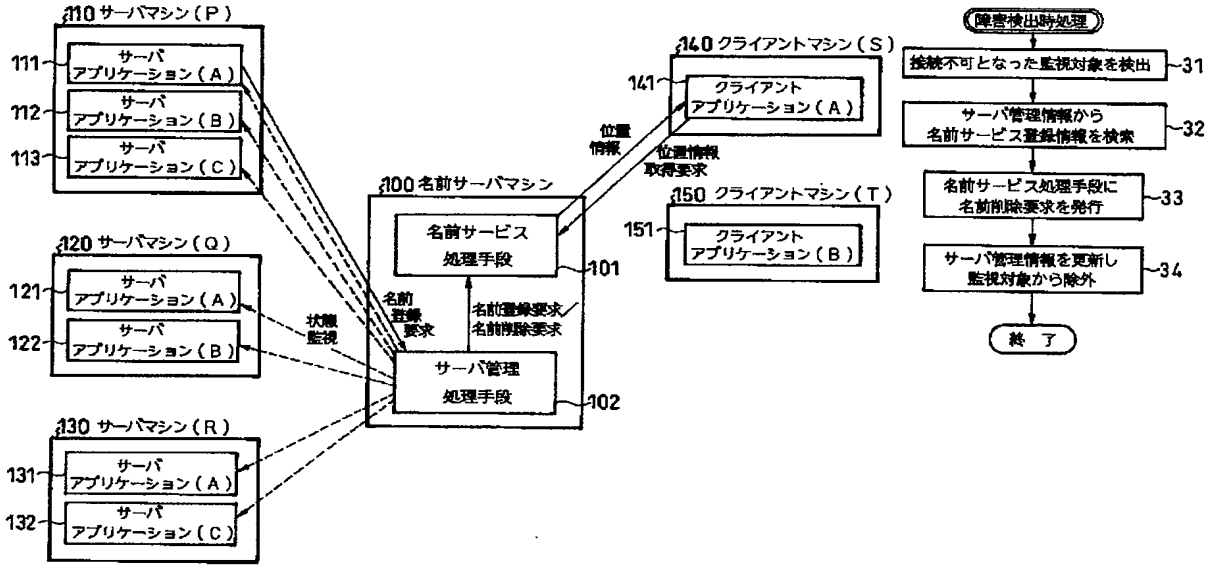
【図１１】本発明の第２の実施の形態に係るサーバアプリケーションの管理選択方式の構成を示すブロック図である。

【符号の説明】

１００　名前サーバマシン
１０１　名前サービス処理手段
１０２　サーバ管理処理手段
１１０　サーバマシン（Ｐ）
１１１，１２１，１３１　サーバアプリケーション（Ａ）
１１２，１２２　サーバアプリケーション（Ｂ）
１１３，１３２　サーバアプリケーション（Ｃ）
１２０　サーバマシン（Ｑ）
１３０　サーバマシン（Ｒ）
１４０　クライアントマシン（Ｓ）
１４１　クライアントアプリケーション（Ａ）
１５０　クライアントマシン（Ｔ）

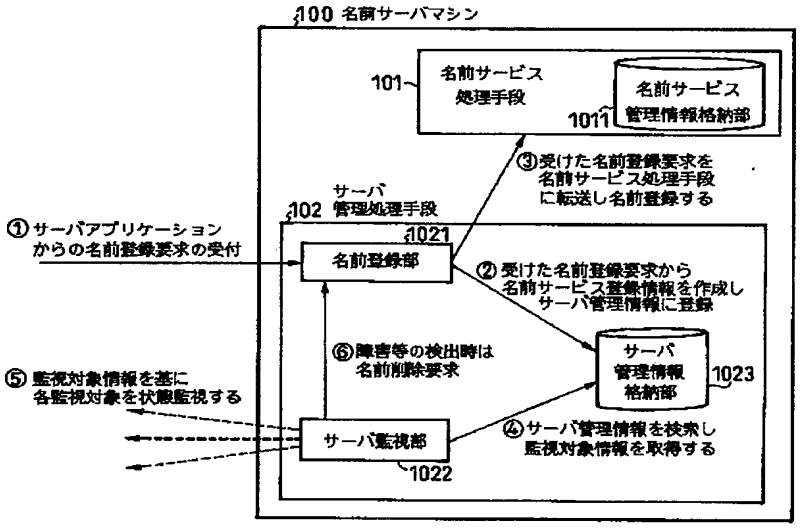
１５１　クライアントアプリケーション（Ｂ）
３１０　監視対象情報
３２０　名前サービス登録情報
４１０　名前
４２０　位置情報
１０１１　名前サービス管理情報格納部
１０２１　名前登録部
１０２２　サーバ監視部
１０２３　サーバ管理情報格納部

【図１】

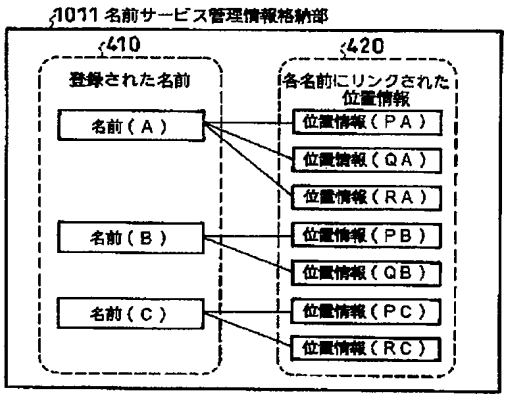


【図６】

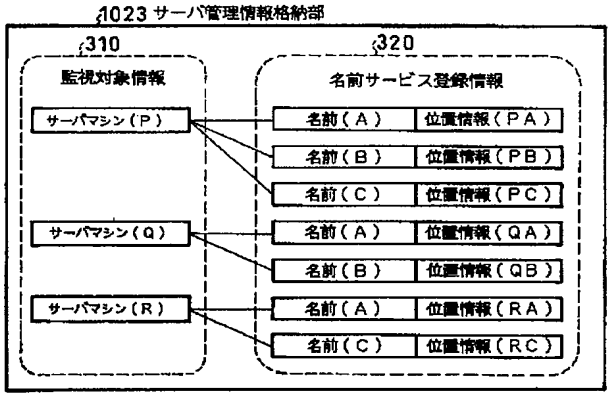
【図２】



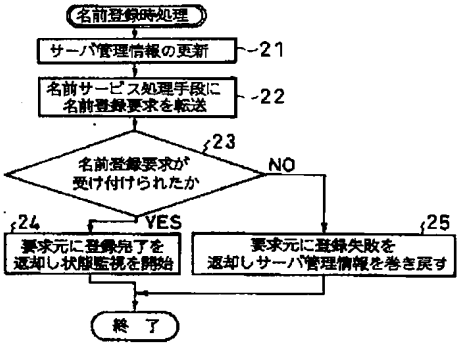
【図４】



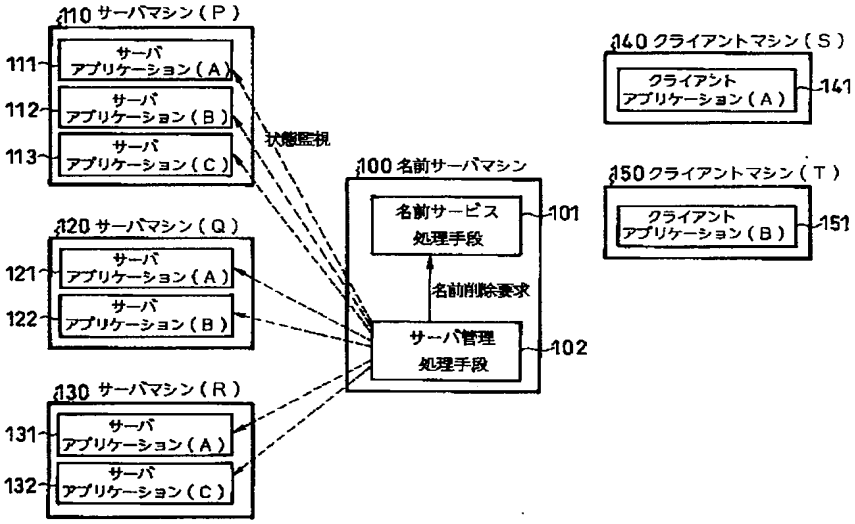
【図３】



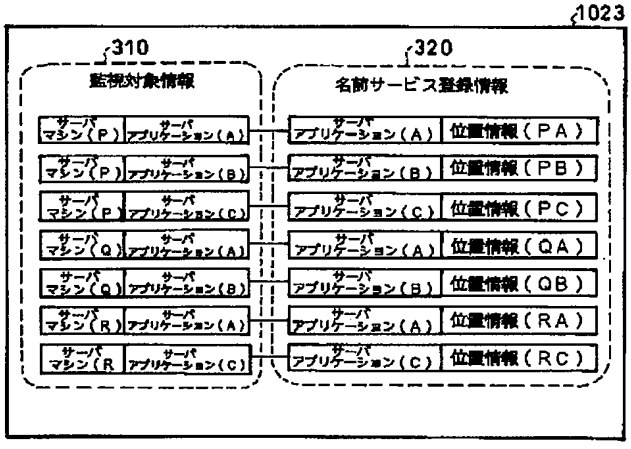
【図５】



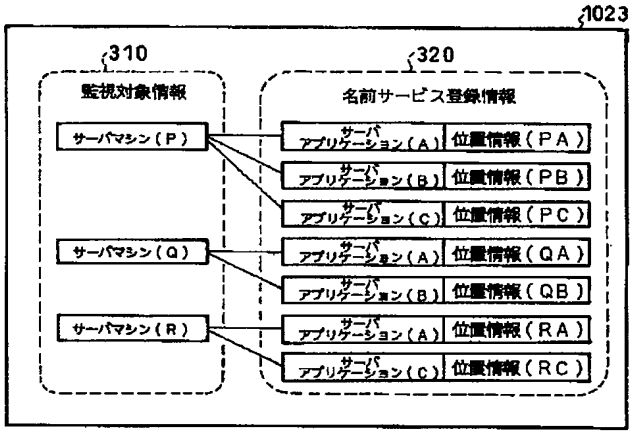
【図７】



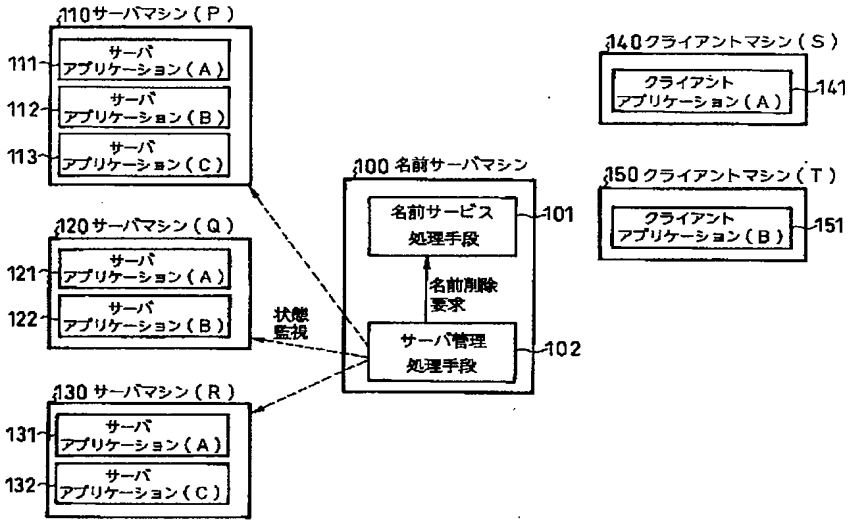
【図８】



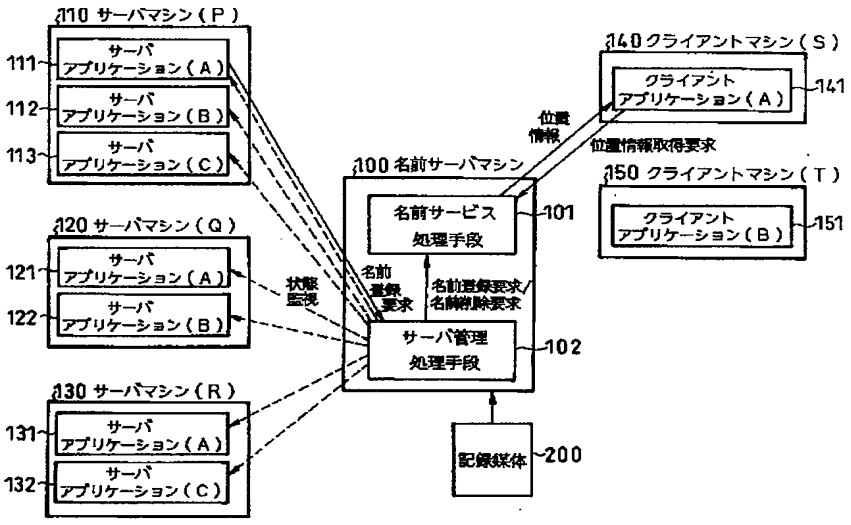
【図１０】



【図９】


110サーバマシン（Ｐ）
111
サーバ
アプリケーション（Ａ）
112
サーバ
アプリケーション（Ｂ）
113
サーバ
アプリケーション（Ｃ）
120サーバマシン（Ｑ）
121
サーバ
アプリケーション（Ａ）
122
サーバ
アプリケーション（Ｂ）
130サーバマシン（Ｒ）
131
サーバ
アプリケーション（Ａ）
132
サーバ
アプリケーション（Ｃ）
状態
監視
100名前サーバマシン
名前サービス
処理手段
101
名前削除
要求
サーバ管理
処理手段
102
140クライアントマシン（Ｓ）
クライアント
アプリケーション（Ａ）
141
150クライアントマシン（Ｔ）
クライアント
アプリケーション（Ｂ）
151

【図１１】


110サーバマシン（Ｐ）
111
サーバ
アプリケーション（Ａ）
112
サーバ
アプリケーション（Ｂ）
113
サーバ
アプリケーション（Ｃ）
120サーバマシン（Ｑ）
121
サーバ
アプリケーション（Ａ）
122
サーバ
アプリケーション（Ｂ）
130サーバマシン（Ｒ）
131
サーバ
アプリケーション（Ａ）
132
サーバ
アプリケーション（Ｃ）
状態
監視
100名前サーバマシン
名前サービス
処理手段
101
名前
登録
要求
名前登録要求/
名前削除要求
サーバ管理
処理手段
102
記録媒体
200
位置
情報
位置情報取得要求
140クライアントマシン（Ｓ）
クライアント
アプリケーション（Ａ）
141
150クライアントマシン（Ｔ）
クライアント
アプリケーション（Ｂ）
151

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2001-209568**
(43)Date of publication of application : **03.08.2001**

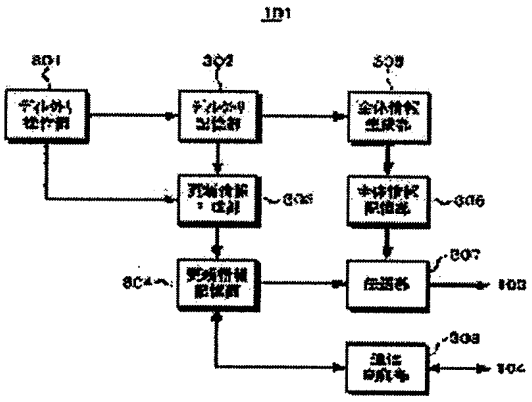
(51)Int.Cl. G06F 12/00
G06F 13/00
// H04N 7/173

(21)Application number : **2000-024769**
(22)Date of filing : **28.01.2000**

(71)Applicant : **JISEDAI JOHO HOSO SYSTEM KENKYUSHO:KK**
SONY CORP
(72)Inventor : **NISHIO IKUHIKO**
GONNO YOSHIHISA
HARAOKA KAZUO
TAKABAYASHI KAZUHIKO
YAMAGISHI YASUAKI

(54) TRANSMITTER, RECEIVER, TRANSMITTER-RECEIVER, METHODS FOR TRANSMISSION AND RECEPTION

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce processing load for receiving in directory service by synchronized approach to total information.
SOLUTION: Information for directory renewal operation is supplied to updating information generation part 303, and updating information for reflecting updating contents to a directory server is generated. The renewed information is then stored to a renewal storage part 304 and transmitted to a network 103 of the information. The total directory information is periodically generated in a part 305 for generating the total information, stored in a total information storage 306 and transmitted to the information network 103. Version number that happens to be renewed immediately below the container is stored in the respective container entries to be stored in a directory storage 302. In addition to the up-to-date contents to be processed, the container



entry name gained at renewal and the version number finally renewed immediately below the container entry are involved in the total information to be created in an area 305 where the total information is generated.

(5)

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2001－209568
（P2001－209568A）
(43)公開日　平成13年８月３日(2001.8.3)

(51)Int.Cl.⁷		識別記号	FI		テーマコード(参考)
Ｇ０６Ｆ	12/00	５４５	Ｇ０６Ｆ 12/00	５４５Ａ	５Ｂ０８２
		５１７		５１７	５Ｂ０８９
	13/00	３５７	13/00	３５７Ｚ	５Ｃ０６４
// Ｈ０４Ｎ	7/173	６１０	Ｈ０４Ｎ 7/173	６１０Ｚ	

審査請求　未請求　請求項の数10　ＯＬ　（全 13 頁）

(21)出願番号　特願2000－24769(P2000－24769)

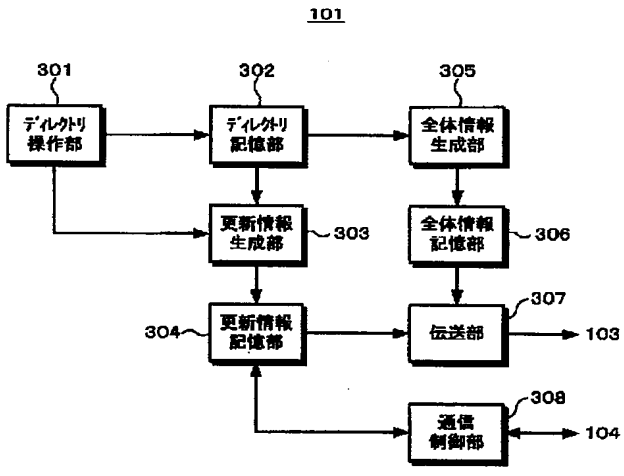
(22)出願日　平成12年１月28日(2000.1.28)

(71)出願人　597136766
株式会社次世代情報放送システム研究所
東京都台東区西浅草１丁目１－１
(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号
(72)発明者　西尾　郁彦
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(74)代理人　100082762
弁理士　杉浦　正知

最終頁に続く

(54)【発明の名称】　送信装置、受信装置、送受信装置、送信方法および受信方法

(57)【要約】
【課題】　ディレクトリサービスにおいて、全体情報で同期をとる時に、受信側の処理の負荷を軽くする。
【解決手段】　ディレクトリの更新操作の情報が更新情報生成部３０３に供給され、更新内容を受信側ディレクトリサーバに反映するための更新情報が生成される。更新情報が更新情報記憶部３０４に記憶され、また、同報ネットワーク１０３に送信される。全体情報生成部３０５では、ディレクトリ全体の情報が定期的に生成され、全体情報が全体情報記憶部３０６に記憶され、同報ネットワーク１０３に送信される。ディレクトリ記憶部３０２に記憶される各コンテナエントリには、そのコンテナエントリの直下のエントリに更新が起こったときのバージョン番号を格納する。全体情報生成部３０５が生成する全体情報には、更新の起こったコンテナエントリ名と更新操作内容に加えて、最後にそのコンテナエントリ直下に更新が起きた時のバージョン番号を含める。



【特許請求の範囲】

【請求項１】　木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、
上記ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、
上記ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、
上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、
上記全体情報および上記更新情報をネットワークにより送信する伝送手段とを備え、
上記全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信装置。

【請求項２】　請求項１において、
上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持つことを特徴とする送信装置。

【請求項３】　請求項１において、
上記ネットワークが同報ネットワークであることを特徴とする送信装置。

【請求項４】　請求項１において、
上記ネットワークが双方向ネットワークであることを特徴とする送信装置。

【請求項５】　木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、
上記ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、
上記ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信する受信手段と、
上記更新情報および上記全体情報の内容に従ってディレクトリを更新する更新手段とを備え、
上記全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、
上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、
上記更新手段は、受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信装置。

【請求項６】　請求項５において、
上記ネットワークが同報ネットワークであることを特徴とする送信装置。

【請求項７】　請求項５において、
上記ネットワークが双方向ネットワークであることを特徴とする送信装置。

【請求項８】　送信装置と受信装置とがネットワークを介して接続され、上記送信装置および受信装置の間で、ディレクトリの同期をとるようにした送受信装置において、
上記送信装置は、
木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、
上記ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、
上記ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、
上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、
上記全体情報および上記更新情報をネットワークにより送信する伝送手段とを備え、
上記全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成するように構成され、
上記受信装置は、
木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、
上記ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、
上記送信装置から上記更新情報および上記全体情報を上記ネットワークを介して受信する受信手段と、
上記更新情報および上記全体情報の内容に従ってディレクトリを更新する更新手段とを備え、
上記更新手段は、受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映するように構成されることを特徴とする送受信装置。

【請求項９】　ディレクトリ記憶手段に記憶され、木構造のデータ構造をもったディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに変化するバージョン番号を管理するステップと、
上記ディレクトリの更新内容を表す更新情報を生成するステップと、

上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成するステップと、
上記全体情報および上記更新情報をネットワークにより送信するステップとからなり、
上記全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信方法。
【請求項１０】　ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信するステップと、
上記更新情報および上記全体情報の内容に従ってディレクトリを更新するステップとを備え、
上記全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、
上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、
受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】この発明は、例えば不特定多数へのデータ配信の技術分野で用いられ、配信されたデータを効率良く更新管理するための送信装置、受信装置、送受信装置、送信方法および受信方法に関する。
【０００２】
【従来の技術】ディレクトリサービスの規格として、Ｘ．５００やＬＤＡＰ(Lightweight Directory Access Protocol) が知られている。Ｘ．５００は、ＩＴＵ－Ｔ(International Telecommunication Union-Telecommunication Standardization Sector)で勧告されているディレクトリサービスの国際標準であり、ディレクトリの概念や、階層構造、サービス、オブジェクトの定義などが規定されている。
【０００３】Ｘ．５００は、汎用性の高いディレクトリサービスプロトコルである反面、実装が重くなる仕様となっていおり、その実装コストが高い。このため、これを、インターネット上のWWW(World Wide Web)ブラウザや、電子メールソフトウェアなどからも簡単に利用することができるように、軽量化、簡素化したサービスプロトコルとして、ＬＤＡＰが開発された。
【０００４】ＬＤＡＰは、サーバ・クライアント型のモデルで構成されており、ディレクトリ情報を格納するサーバ（ディレクトリサーバ）と、それにアクセスするクライアントとの間のリクエスト・レスポンス型のメッセージ転送プロトコルを定義している。ディレクトリサーバで集中管理されるディレクトリ情報の規模が大きくなると、クライアントのアクセスが集中することによって、サーバの質が低下するために、複数のディレクトリサーバによって、ディレクトリ情報が分散管理される。
【０００５】ディレクトリ情報が分散管理されている場合、あるディレクトリサーバが管理しているエントリ（ディレクトリ情報の最小管理単位）の複製が他の１以上のディレクトリサーバに存在することになる。したがって、あるディレクトリサーバにおいてエントリの更新が生じた時には、他のディレクトリサーバにおいても、複製を更新する必要がある。このように、エントリの内容が異なるものとならないように、同期管理が必要とされる。すなわち、あるディレクトリサーバが管理しているエントリが更新された場合に、他のディレクトリサーバが管理しているそのエントリの複製の更新を即座に行うことが必要とされる。同期管理方式として、同報ネットワークを使用した放送型プロトコルを主体としたものが先に提案されている（特願平１０－２７７３５２号および特願平１０－２７７３５３号参照）。
【０００６】
【発明が解決しようとする課題】上述した同期管理方式を有するディレクトリサービスシステムにおいて、更新内容は、更新前と更新後の差分のデータとして伝送される。若し、受信側のディレクトリサーバが電源オフ等のために長い間休止していると、連続した差分更新通知メッセージを取りこぼすおそれが高くなる。このような場合、放送によって伝送される差分情報だけでは、受信側のディレクトリを同期させることができなくなる。その結果、受信側は、差分情報ではなく、既に受信しているディレクトリ情報を捨て、ディレクトリ全体の情報を最初から受信してディレクトリ全体を再構築することによって、同期した状態へ回復することになる。一般的にディレクトリ全体の情報は、差分に比べて非常に大きなデータサイズとなるため、ディレクトリ回復作業が受信側にとって負荷が高い処理となる。
【０００７】従って、この発明の目的は、全体の情報を受信することによって受信側が送信側とディレクトリを同期させる際に、既に受信済のディレクトリ構造を可能な限り保持しながらディレクトリ同期を行い、それによって処理の負荷を軽減することが可能な送信装置、受信装置、送受信装置、送信方法および受信方法を提供することにある。
【０００８】
【課題を解決するための手段】上述した課題を解決するために、請求項１の発明は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新す

ると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、全体情報および更新情報をネットワークにより送信する伝送手段とを備え、全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信装置である。

【０００９】請求項５の発明は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信する受信手段と、更新情報および全体情報の内容に従ってディレクトリを更新する更新手段とを備え、全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、更新手段は、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信装置である。

【００１０】請求項８の発明は、送信装置と受信装置とがネットワークを介して接続され、送信装置および受信装置の間で、ディレクトリの同期をとるようにした送受信装置において、送信装置は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、全体情報および更新情報をネットワークにより送信する伝送手段とを備え、全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成するように構成され、受信装置は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、送信装置から更新情報および全体情報をネットワークを介して受信する受信手段と、更新情報および全体情報の内容に従ってディレクトリを更新する更新手段とを備え、更新手段は、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映するように構成されることを特徴とする送受信装置である。

【００１１】請求項９の発明は、ディレクトリ記憶手段に記憶され、木構造のデータ構造をもったディレクトリを更新すると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するステップと、ディレクトリの更新内容を表す更新情報を生成するステップと、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成するステップと、全体情報および更新情報をネットワークにより送信するステップとからなり、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信方法である。

【００１２】請求項１０の発明は、ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信するステップと、更新情報および全体情報の内容に従ってディレクトリを更新するステップとを備え、全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信方法である。

【００１３】この発明では、各エントリに対して、その直下に更新が起きた時のバージョン番号を付加することによって、全体情報によってディレクトリの同期を取り直す時に、必要な部分のみを処理することができる。

【００１４】

【発明の実施の形態】以下、この発明の一実施形態について説明する。図１は、一実施形態におけるデータ配信システムの一例を示す。送信側ディレクトリサーバ１０１は、ディレクトリを保持、管理している。ディレクトリは、木構造のデータ構造を有し、その最小管理単位がコンテナエントリ（単にエントリとも言う）であり、また、コンテナエントリの追加、削除を行うことができる。一例として、ディレクトリサーバ１０１には、情報提供者（図示しない）から提供された交通情報、天気情

報、株価情報等のリアルタイムで変化するデータ、リアルタイムで変化しないデータ、テキストデータ、画像データ、オーディオデータ、コンピュータプログラム等のデータが記憶される。ディレクトリのひとかたまりのデータがコンテンツ、またはオブジェクトと呼ばれる。また、送信側ディレクトリサーバ１０１の一例は、放送局である。

【００１５】送信側ディレクトリサーバ１０１は、同報ネットワーク（例えば放送ネットワーク）１０３および双方向ネットワーク１０４を介して受信側ディレクトリサーバ102_1，102_2にディレクトリを提供する。双方向ネットワーク１０４は、具体的には、アナログ公衆電話網、ＩＳＤＮ(Integrated Services Digital Network)、インターネット等である。受信側ディレクトリサーバの個数は、実際には、無数あるが、図１は、代表的に２個の受信側ディレクトリサーバ102_1，102_2を示している。

【００１６】受信側ディレクトリサーバ102_1，102_2は、送信側ディレクトリサーバ１０１から提供されるディレクトリを受信し、蓄積し、利用者に提供するものである。そのため、送信側ディレクトリサーバ１０１に記憶されるディレクトリの全て、またはその一部を同報ネットワーク１０３および双方向ネットワーク１０４を介して受信し、記憶（蓄積）する。

【００１７】また、送信側ディレクトリサーバ１０１において起こったディレクトリの更新は、同報ネットワーク１０３および双方向ネットワーク１０４を介して受信側ディレクトリサーバ102_1，102_2に反映される。それによって、受信側ディレクトリサーバ１０１と送信側ディレクトリサーバ102_1，102_2の間でディレクトリの内容の同期が取られる。

【００１８】ディレクトリは、図２に示すように、木構造で管理されるデータである。木構造を構成する各ノードのうちさらにその下にノードをもつものがコンテナエントリ２０１であり、一番下に位置するノードがリーフノード２０２である。図２中、Ｃ１，Ｃ２，・・・，Ｃ４がコンテナエントリであり、Ｌ１，Ｌ２，Ｌ３がリーフノードである。例えば放送番組表の場合では、コンテナエントリＣ２がニュース番組であり、リーフノードＬ１が午後９時に放送されるニュース番組である。勿論、職業別電話帳等も図２に示す木構造で表すことができる。

【００１９】図３は、送信側ディレクトリサーバ１０１の構成例を示す。ディレクトリ記憶部３０２には、図２に示す構造を持ったディレクトリが記憶される。ディレクトリ記憶部３０２にデータを追加したり、記憶されたデータを削除するのがディレクトリ操作部３０１である。ディレクトリ操作部３０１によるディレクトリの更新操作の情報が更新情報生成部３０３に供給される。更新操作の情報を受けて、更新情報生成部３０３では、ディレクトリの更新内容を受信側ディレクトリサーバ102_1，102_2のディレクトリに反映するための更新情報が生成される。

【００２０】更新情報生成部３０３によって生成された更新情報が更新情報記憶部３０４に記憶される。更新情報記憶部３０４には、受信側での更新情報の取りこぼしを想定して一定期間繰り返して更新情報を送信するために、複数の更新情報が記憶される。更新情報記憶部３０４に記憶された更新情報は、伝送部３０７によって同報ネットワーク１０３に繰り返して送信される。

【００２１】ディレクトリ記憶部３０２には、全体情報生成部３０５が接続される。全体情報生成部３０５では、受信側ディレクトリサーバ102_1，102_2での更新情報の取りこぼしを考慮し、ディレクトリ記憶部３０２に記憶されたディレクトリ全体の情報が定期的に生成される。生成された全体情報が全体情報記憶部３０６に記憶され、伝送部３０７から同報ネットワーク１０３に一定期間、繰り返して送信される。

【００２２】双方向ネットワーク１０４に通信制御部３０８が接続される。双方向ネットワーク１０４を介して受信された受信側ディレクトリサーバ102_1，102_2からの要求に従って、更新情報記憶部３０４に記憶された更新情報またはディレクトリ記憶部３０２に記憶されるディレクトリ全体を双方向ネットワーク１０４を介して受信側ディレクトリサーバ102_1，102_2に提供する。

【００２３】図４は、受信側ディレクトリサーバ102_1，102_2の構成例を示す。同報ネットワーク１０３に対して受信部４０１が接続される。受信部４０１が同報ネットワーク１０３を介して送信側ディレクトリサーバ１０１からディレクトリの更新情報および全体情報を受信する。受信した情報は、更新操作部４０３において解釈され、ディレクトリ記憶部４０４に記憶されているディレクトリに反映される。全体情報を受信したのであれば、そのデータをディレクトリ記憶部４０４に記憶する。また、更新情報を受信したのであれば、ディレクトリ記憶部４０４に記憶されているディレクトリの指示された部分を更新する。

【００２４】双方向ネットワーク１０４に対して通信制御部４０２が接続される。通信制御部４０２は、受信部４０１で取りこぼした全体情報または更新情報を送信側ディレクトリサーバ１０１に要求する。そして、受信した全体情報や更新情報を更新操作部４０３を通じてディレクトリ記憶部４０４に記憶されているディレクトリに反映する。ディレクトリ操作部４０５は、ディレクトリ記憶部４０４に記憶されているディレクトリを閲覧したり、参照するための操作を行うためのものである。

【００２５】図５を参照して、送信側ディレクトリサーバ１０１におけるディレクトリの更新例について説明する。更新操作は、ディレクトリ操作部３０１によって行

われる。まず、コンテナエントリＣ０にＣ１、Ｃ２が順に追加される（データ構造５０１）。次に、コンテナエントリＣ３がコンテナエントリＣ１の下に追加される（データ構造５０２）。コンテナエントリＣ２の下にコンテナエントリＣ４が追加される（データ構造５０３）。コンテナエントリＣ２の下にコンテナエントリＣ５が追加される（データ構造５０４）。最後にコンテナエントリＣ２の下からコンテナエントリＣ４が削除される（データ構造５０５）。

【００２６】図５に示すようなディレクトリの更新が起こった場合を考えると、送信側ディレクトリサーバ１０１の更新情報生成部３０３および全体情報生成部３０５では、図６に示すような全体情報および更新情報が生成される。これらの全体情報および更新情報が同報ネットワーク１０３を介して送信される。また、受信側ディレクトリサーバ102_1，102_2の要求に応じて、要求された全体情報または更新情報が双方向ネットワーク１０４を介して送信される。

【００２７】一実施形態においては、ディレクトリ記憶部３０２に記憶される各コンテナエントリには、そのコンテナエントリの直下のエントリに更新が起こったときのスキーマのバージョン（バージョン番号と呼ぶことにする）を格納する。すなわち、コンテナエントリの状態は、（コンテナエントリ名，バージョン番号）で表すことができる。バージョン番号は、更新のたびに変化し、更新の順序を示す値であれば、昇順の数、降順の数等、種々の値を使用できる。

【００２８】全体情報生成部３０５が生成する全体情報には、更新の起こったコンテナエントリ名と更新操作内容に加えて、最後にそのコンテナエントリ直下に更新が起きた時のバージョン番号を含める。つまり、全体情報は、（バージョン番号，コンテナエントリ名，更新操作）で表される。更新情報も同様に表す。操作は、コンテナエントリの追加、または削除である。

【００２９】図６中の全体情報について説明すると、最初のバージョン番号（０．３）がデータ構造５０１に対応したものである。すなわち、仮想的なコンテナエントリ（またはノード）ｒにコンテナエントリＣ０を追加したものがバージョン番号（０．１）の全体情報（０．１，ｒ，＋Ｃ０）である。バージョン番号（０．１）に対して、コンテナエントリＣ０にＣ１を追加したバージョン番号が（０．２）である。さらに、コンテナエントリＣ２を追加した全体情報が（０．３，Ｃ０，＋Ｃ１）である。

【００３０】コンテナエントリＣ０の直下に最後に更新が起きた時のバージョン番号は、コンテナエントリＣ２が追加された時のバージョン番号（０．３）であるので、コンテナエントリＣ１を追加した時の全体情報が（０．３，Ｃ０，＋Ｃ１）とされる。以上の３個の全体情報によって、データ構造５０１が復元できる。また、更新情報としての差分情報は、バージョン番号（０．２）からバージョン番号（０．３）に変化した時の変化を示すように、コンテナエントリＣ０の直下に最後に更新が起きた時のバージョン番号（０．３）と、更新操作からなる。すなわち、（０．３，Ｃ０，＋Ｃ２）である。（＋）が追加を表す。

【００３１】次に、コンテナエントリＣ１に対して、コンテナエントリＣ３を追加してデータ構造５０２（バージョン番号（１．０））が生成される。この場合には、コンテナエントリＣ１の直下に最後に更新が起きた時のバージョン番号が（１．０）であるので、（１．０，Ｃ１，＋Ｃ３）の全体情報が生成される。コンテナエントリＣ０についてのバージョン番号は、（０．３）のままである。また、差分情報が（１．０，Ｃ１，＋Ｃ３）となる。

【００３２】次に、コンテナエントリＣ２に対して、コンテナエントリＣ４を追加してデータ構造５０３（バージョン番号（１．１））が生成される。この場合には、コンテナエントリＣ２の直下に最後に更新が起きた時のバージョン番号が（１．１）であるので、（１．１，Ｃ２，＋Ｃ４）の全体情報が生成される。コンテナエントリＣ０についてのバージョン番号は、（０．３）のままで、コンテナエントリＣ１についてのバージョン番号は、（１．０）のままである。また、差分情報が（１．１，Ｃ２，＋Ｃ４）となる。

【００３３】次に、コンテナエントリＣ２に対して、コンテナエントリＣ５を追加してデータ構造５０４（バージョン番号（１．２））が生成される。この場合には、コンテナエントリＣ２の直下に最後に更新が起きた時のバージョン番号が（１．２）であるので、（１．２，Ｃ２，＋Ｃ５）の全体情報が生成される。これと共に、バージョン番号（１．１）では、（１．１，Ｃ２，＋Ｃ４）であった全体情報が（１．２，Ｃ２，＋Ｃ４）に変更される。また、差分情報が（１．２，Ｃ２，＋Ｃ５）となる。

【００３４】最後に、コンテナエントリＣ２に対して追加されたコンテナエントリＣ４が削除されてデータ構造５０５（バージョン番号（１．３））が生成される。この場合には、コンテナエントリＣ２の直下に最後に更新が起きた時のバージョン番号が（１．３）であるので、（１．３，Ｃ２，＋Ｃ５）の全体情報が生成される。これと共に、コンテナエントリＣ４の削除の操作と対応して、全体情報（１．２，Ｃ２，＋Ｃ４）が削除される。また、差分情報が（１．３，Ｃ２，－Ｃ４）となる。（－）が削除を表す。

【００３５】図７は、送信側ディレクトリサーバ１０１において、更新操作がなされた時になされる各コンテナエントリＣｎのバージョン番号管理の流れを示すフローチャートである。ステップＳ１において、あるコンテナエントリＣｎの直下に更新（コンテナエントリの追加ま

たは削除）が起こると、ステップＳ２において、バージョン番号Ｖを１増加する。そして、ステップＳ３において、＋１で得られたバージョン番号Ｖをコンテナエントリ Ｃｎの直下に更新が起きた時のバージョン番号Ｃｎｖとして記憶する。

【００３６】図８は、送信側ディレクトリサーバ１０１における全体情報の生成処理の流れを示すフローチャートである。コンテナエントリの総数をＮとし、処理の対象とするコンテナエントリがｉ番目のコンテナエントリＣｉとすると、ステップＳ１１では、（ｉ＜Ｎ）かどうかが決定される。（ｉ＜Ｎ）の関係が成立するときは、全てのコンテナエントリの処理が終了していないことを意味するので、処理がステップＳ１２に移る。若し、（ｉ＜Ｎ）の関係が成立しないときは、全てのコンテナエントリに関する処理終了したものと決定され、処理が終了する。

【００３７】ステップＳ１２においては、コンテナエントリＣｉの直下にある全てのコンテナエントリＣｊに関して処理がなされる。すなわち、ステップＳ１３において、全体情報に付加するバージョン番号ｖをコンテナエントリＣｉの記憶しているバージョン番号Ｃｉｖとし（ステップＳ１３）、次のステップＳ１４において、全体情報（ｖ，Ｃｉ，＋Ｃｊ）、すなわち、（Ｃｉｖ，Ｃｉ，＋Ｃｊ）を生成する。

【００３８】全てのコンテナエントリＣｊについての処理（ステップＳ１３およびステップＳ１４）が終了すると、ステップＳ１５に処理が移り、コンテナエントリ番号ｉが１増加される。そして、ステップＳ１１に戻り、全てのコンテナエントリＣｉについての処理が終了したか否かが決定される。

【００３９】ここで、受信側ディレクトリサーバ102_1，102_2が長期間、休止状態であった後に復帰した場合、ディレクトリ記憶部４０４に記憶されている情報を最新の状態に同期させる既存の方法について説明する。まず、ディレクトリの全体情報を受信する。ディレクトリの全体情報を受信し、ディレクトリを同期する際に、既にディレクトリ記憶部４０４に記憶されている情報を全て破棄し、改めて全体情報からディレクトリを再構築する。しかしながら、このような既存の同期方法は、受信側ディレクトリサーバにとって、処理の負荷が重い。

【００４０】この発明では、ディレクトリを送信側ディレクトリサーバと同期させるために、既に受信済の情報に関して変化が起きたか否かを先に判定することによって処理の負荷を軽減することが可能なものである。その判定のために、送信側ディレクトリサーバ１０１において、上述したように生成された全体情報中のバージョン番号、すなわち、各コンテナエントリに付加されている、そのコンテナエントリの直下のエントリに更新が起こったときのバージョン番号を使用する。

【００４１】図９は、受信側ディレクトリサーバ102_1，102_2の更新処理の流れを示すフローチャートである。最初のステップＳ２１において、最新の差分更新情報に含まれるバージョン番号Ｖｄと受信側に記憶されているバージョン番号Ｖｓとを比較する。（Ｖｄ＝Ｖｓ＋１）ならば、取りこぼしがなく、正常に差分更新情報を受信していることが分かる。その場合には、ステップＳ２２において、差分更新を適用し、バージョン番号Ｖｓを１増加し、更新処理を終了する。ステップＳ２２の処理によって、受信側のバージョン番号が送信側のバージョン番号と等しくなる。

【００４２】ステップＳ２１において、（Ｖｄ＝Ｖｓ＋１）でなければ、すなわち、（Ｖｄ＞Ｖｓ＋１）ならば、差分更新情報の取りこぼしが起きているものと決定される。この場合では、全体情報を受信することによって、復旧が図られる。まず、ステップＳ２３において、全ての全体情報を受信したか否かが決定される。全ての全体情報を受信していれば、復旧が完了する。

【００４３】ステップＳ２３において、全ての全体情報を受信していないならば、ステップＳ２４において、全体情報（ｖ，Ｃｉ，＋Ｃｊ）を受信する。ステップＳ２５では、全体情報中のバージョン番号ｖと受信側に記憶されているバージョン番号Ｖｓとの関係が（ｖ＞Ｖｓ）か否かが決定される。（ｖ＞Ｖｓ）でなければ、処理がステップＳ２３に戻る。（ｖ＞Ｖｓ）であれば、コンテナエントリＣｉの直下に不整合が起きていることが分かるので、ステップＳ２６において、コンテナエントリＣｉの直下の全てのコンテナエントリが削除済か否かが決定される。これは、不整合によってコンテナエントリＣｉの直下のコンテナエントリが更新を正しく反映していないおそれがあるからである。

【００４４】削除済でないならば、ステップＳ２７において、コンテナエントリＣｉの直下のコンテナエントリを全て削除し、ステップＳ２８に処理が移る。ステップＳ２６において、削除済と決定されたときも、処理がステップＳ２８に移る。ステップＳ２８では、全体情報（ｖ，Ｃｉ，＋Ｃｊ）の内容をコンテナエントリＣｉの直下に反映するために、コンテナエントリＣｉにコンテナエントリＣｊを追加する。そして、処理がステップＳ２３に戻る。全ての全体情報を受信するまで、ステップＳ２４～Ｓ２８の処理が繰り返される。

【００４５】図６の例を参照してより具体的に説明すると、例えば受信側ディレクトリサーバがバージョン番号（１．１）の状態（全体情報（１．１，Ｃ２，＋Ｃ４））の後、バージョン番号（１．２）の間、電源切断等の何らかの理由によって休止状態となったものとする。

【００４６】この場合には、全体情報（１．２，Ｃ２，＋Ｃ４）を受信すると、ステップＳ２５（図９）の関係（ｖ＞Ｖｓ）が満たされ、コンテナエントリＣ２の直下

のエントリＣ４が削除され（ステップＳ２７）、Ｃ２の直下にＣ４が追加される（ステップＳ２８）。そして、次の全体情報（１．２，Ｃ２，＋Ｃ５）を受信すると、コンテナエントリＣ５がＣ２に追加される。このようにして、コンテナエントリＣ２の部分のみを処理することで、ディレクトリの同期なしうる。

【００４７】この発明を適用しない場合、受信側ディレクトリサーバ102_1，102_2は、ディレクトリ記憶部４０４に記憶されるバージョン番号（１．１）のディレクトリ構造を全て破棄し、バージョン（１．３）の全体情報を受信してディレクトリの同期をとることになる。しかしながら、実際には、バージョン（１．１）と（１．３）の間のディレクトリの違いは、Ｃ４、Ｃ５の部分のみであり、その他の部分については同期処理を必要としない。この発明によれば、コンテナエントリＣ２以下の部分についてのみ同期処理を行うことになるので、受信側の処理の負荷を軽くすることができる。

【００４８】この発明は、上述した実施形態等に限定されるものでは無く、この発明の要旨を逸脱しない範囲内で様々な変形や応用が可能である。

【００４９】

【発明の効果】この発明では、ディレクトリの更新において、各コンテナエントリに対して、その直下に更新処理が起きたときのバージョン番号を付加することにより、全体情報によって同期を取り直す時に、必要な部分のみを処理することが可能となり、受信側の処理の負荷を軽くすることができる。

【図面の簡単な説明】

【図１】この発明の一実施形態である、ディレクトリサービスシステムの構成の一例を示す略線図である。

【図２】この発明の一実施形態におけるディレクトリの木構造を説明するための略線図である。

【図３】この発明の一実施形態における送信側ディレクトリサーバの一例を示すブロック図である。

【図４】この発明の一実施形態における受信側ディレクトリサーバの一例を示すブロック図である。

【図５】ディレクトリエントリの更新例を順に示す略線図である。

【図６】受信側ディレクトリサーバにおけるディレクトリ更新処理の一例における全体情報および更新情報を示す略線図である。

【図７】この発明の一実施形態における、送信側でのコンテナエントリの更新バージョン番号管理の処理を示すフローチャートである。

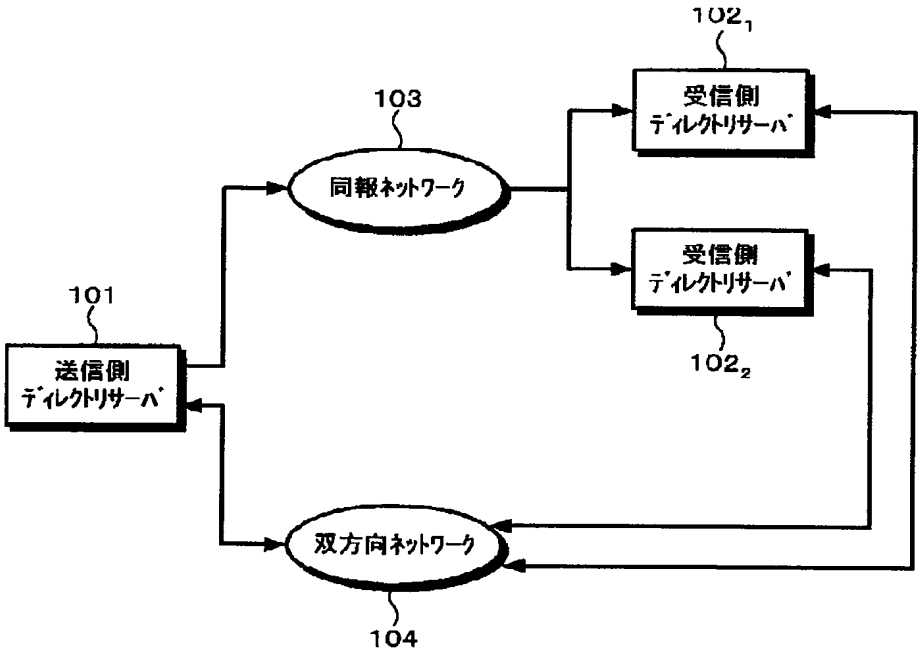
【図８】この発明の一実施形態における、送信側での全体情報の生成処理を示すフローチャートである。

【図９】この発明の一実施形態における、受信側でのディレクトリの更新処理を示すフローチャートである。

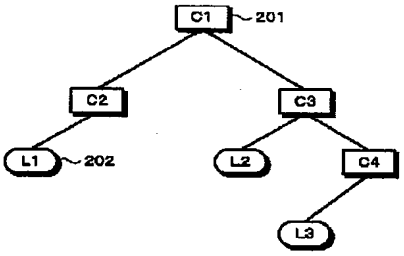
【符号の説明】

１０１・・・送信側ディレクトリサーバ、102_1，102_{21}，102_1・・・受信側ディレクトリサーバ、１０３・・・同報ネットワーク、２０１・・・コンテナエントリ、２０２・・・リーフノード、３０２・・・ディレクトリ記憶部、３０３・・・更新情報生成部、３０４・・・更新情報記憶部、４０１・・・受信部、４０４・・・ディレクトリ記憶部

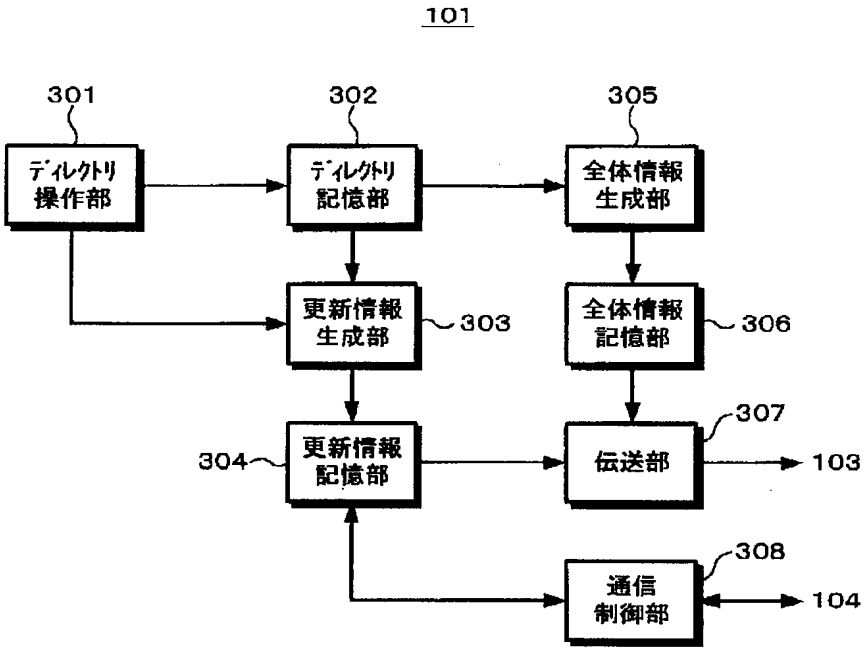
【図１】



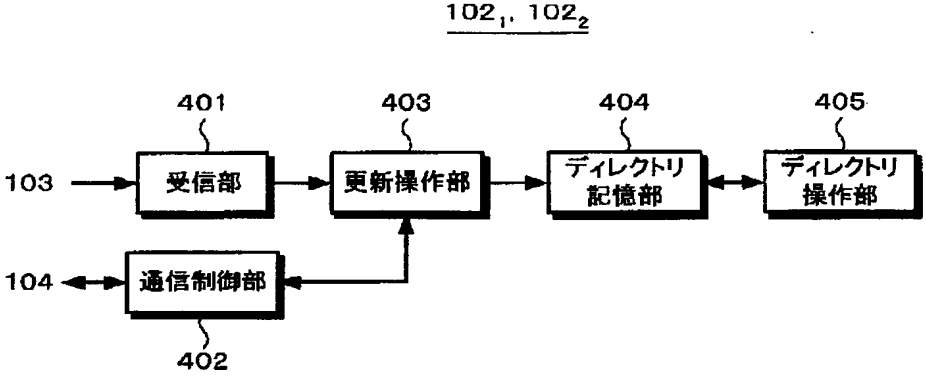
【図２】


C1
201
C2
C3
L1
202
L2
C4
L3

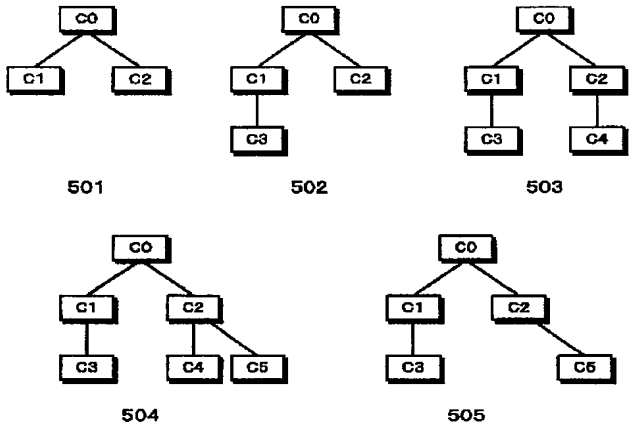
【図３】


101
301
302
305
ディレクトリ操作部
ディレクトリ記憶部
全体情報生成部
更新情報生成部
303
全体情報記憶部
306
307
304
更新情報記憶部
伝送部
103
308
通信制御部
104

【図４】


102_1, 102_2
401
403
404
405
103
受信部
更新操作部
ディレクトリ記憶部
ディレクトリ操作部
104
通信制御部
402

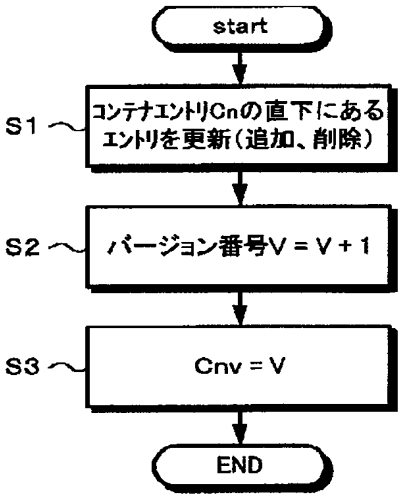
【図５】



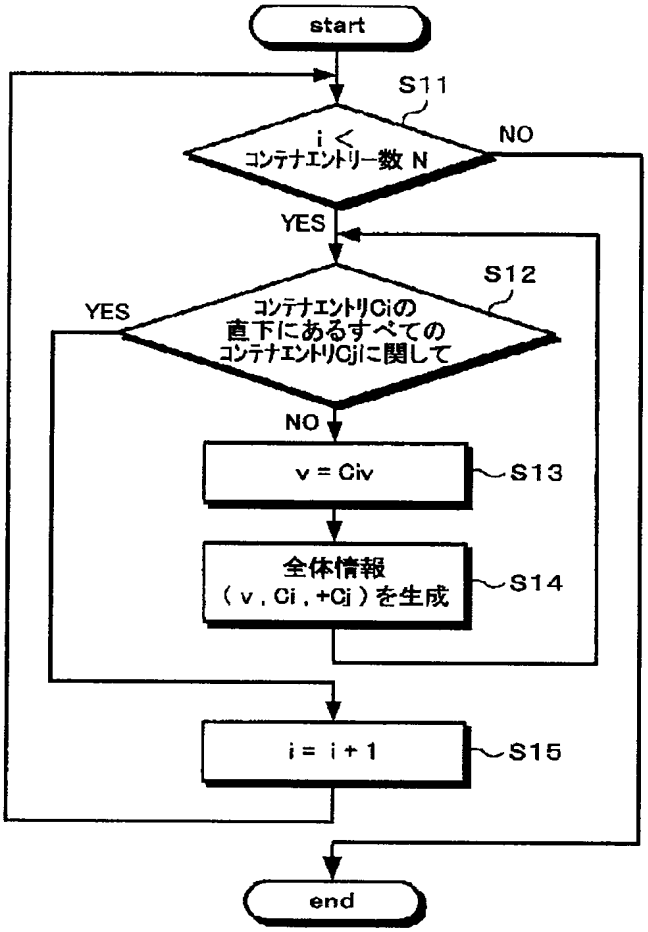
【図６】

ver.	全体情報	差分情報
0.3	(0.1 , r ,+C0) (0.3 ,C0 ,+C1) (0.3 ,C0 ,+C2)	(0.3 , C0 ,+C2)
1.0	(0.1 , r ,+C0) (0.3 ,C0 ,+C1) (0.3 ,C0 ,+C2) (1.0 ,C1 ,+C3)	(1.0 , C1 ,+C3)
1.1	(0.1 , r ,+C0) (0.3 ,C0 ,+C1) (0.3 ,C0 ,+C2) (1.0 ,C1 ,+C3) (1.1 ,C2 ,+C4)	(1.1 , C2 ,+C4)
1.2	(0.1 , r ,+C0) (0.3 ,C0 ,+C1) (0.3 ,C0 ,+C2) (1.0 ,C1 ,+C3) (1.2 ,C2 ,+C4) (1.2 ,C2 ,+C5)	(1.2 , C2 ,+C5)
1.3	(0.1 , r ,+C0) (0.3 ,C0 ,+C1) (0.3 ,C0 ,+C2) (1.0 ,C1 ,+C3) (1.3 ,C2 ,+C5)	(1.3 , C2 ,−C4)

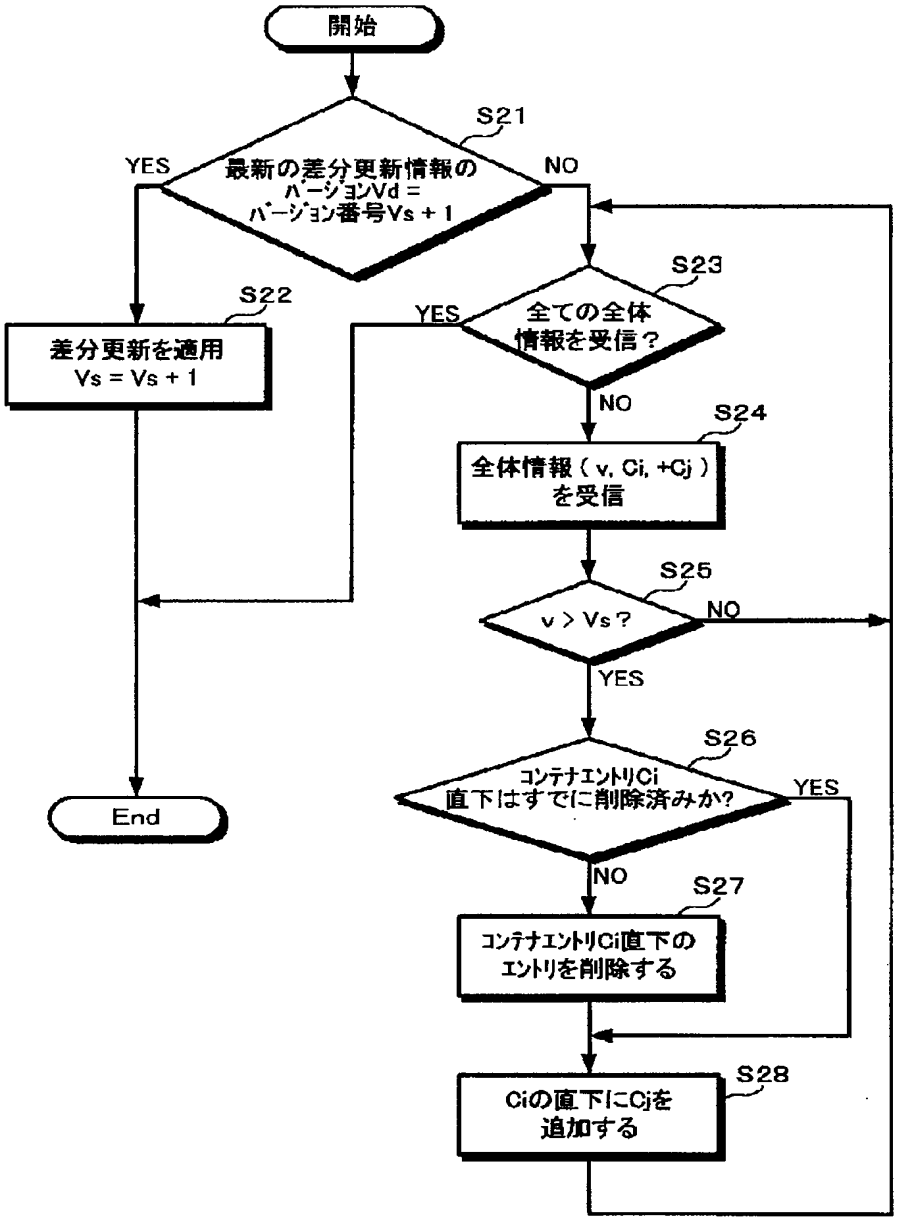
【図７】



【図８】


start
S11
i <
コンテナエントリー数 N
NO
YES
S12
コンテナエントリCiの
直下にあるすべての
コンテナエントリCjに関して
YES
NO
v = Civ
S13
全体情報
(v , Ci , +Cj) を生成
S14
i = i + 1
S15
end

【図９】



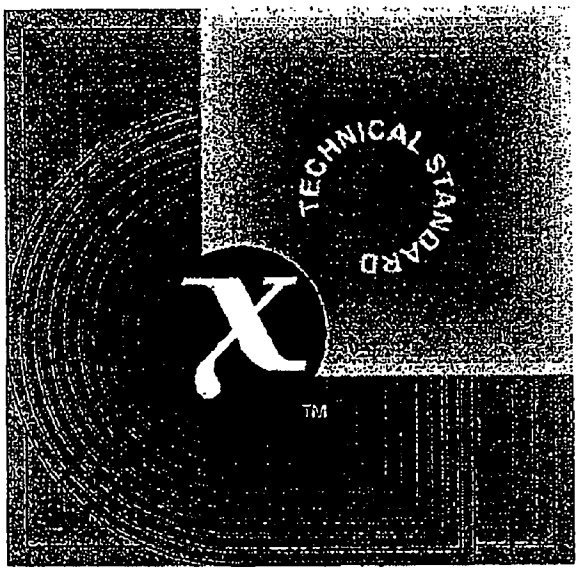
フロントページの続き

(72)発明者　権野　善久
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者　原岡　和生
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者　高林　和彦
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者　山岸　靖明
東京都品川区北品川６丁目７番35号　ソニー株式会社内

Ｆターム(参考) 5B082 FA00 GA15
5B089 GA11 JA11 JB22 KA06 KB10
KC30 KE07
5C064 BB05 BC06 BC10 BC16 BC20
BD01 BD07

Technical Standard

Federated Naming: The XFN Specification



THE Open GROUP

[This page intentionally left blank]

X/Open CAE Specification

Federated Naming: The XFN Specification

X/Open Company Ltd.

X/Open CAE Specification

Federated Naming: The XFN Specification

ISBN: 1-85912-052-0

X/Open Document Number: C403

Published by X/Open Company Ltd., U.K.

Any comments relating to the material contained in this document may be submitted to X/Open at:

X/Open Company Limited
Apex Plaza
Forbury Road
Reading
Berkshire, RG1 1AX
United Kingdom

or by Electronic Mail to:

XoSpecs@xopen.org

2.3 XFN Usage and Implementation Models

This section describes how the XFN interface can be used and implemented. The usage model is from a client's perspective. The implementation model is from a service provider's perspective.

2.3.1 Basic Usage Model

Most clients of the XFN interface will only be interested in lookups. Their usage of the XFN interface will amount essentially to:

- obtaining the Initial Context
- looking up one or more names relative to the Initial Context.

Once the client obtains a desired reference, it constructs a client-side representation of the object from the reference. This need not involve code within the application layer. For example, RPC services can provide clients with a means of constructing client-side handles from a composite name for the service, or from a reference containing an RPC address for the service. After getting this handle, the client performs all further operations on the object or service by supplying the handle.

This is the basic model of how the XFN interface is expected to be used. The XFN enterprise policies presented in Appendix D further encourage a bind/lookup model for how services and clients may rendezvous through the use of the naming service.

Applications may also use federated naming services indirectly through other interfaces. For example, consider a UNIX application that obtains a filename that it later supplies to the UNIX *open*() function. If the system provides XFN support for resolution of filenames, the application need not be aware that the strings it deals with are composite names rather than the traditional local pathnames. Some applications can thereby support the use of composite names without modification.

2.3.2 Basic Implementation Model

The XFN specifications do not dictate a specific way of providing support for the XFN interface. The implementor has great flexibility in terms of how to provide implementations of the XFN service for new and existing naming systems in different environments, based on different requirements in those environments. A potential implementor may find the basic model described below useful in order to understand how to provide the required functionality. This model and additional models are described in more detail in Appendix C, which provides the implementor guidelines on how different configurations of XFN with new and existing naming systems might be implemented.

The XFN client interface can be implemented in a layered manner. Figure 2-1 shows examples of this approach for a few existing naming services.

The top layer (labeled XFN API) is the XFN client interface with which callers interact directly. It implements the types of parameters used in the XFN interface.

In the bottom layer (labeled Context Implementation) are implementations of the XFN interface over specific naming services. Each such implementation over a specific naming service is based on knowledge of that naming service and essentially maps the XFN interface onto that service. It is expected that most of these implementations will work over existing naming services using the existing naming service interfaces and protocols.

Alternatively, the bottom layer could be a generic implementation that supports the XFN interface using a client/server model instead of a naming service-specific implementation. For example, there could be an XFN/DCE service to which the client invokes XFN requests. No

particular naming service-specific implementation or generic implementation is a mandatory component of XFN.

The middle layer (labeled XFN Framework) implements operations that span naming systems using interfaces presented in the bottom layer. This layer is responsible for guiding composite name resolution and invoking the proper operations for each context as required.

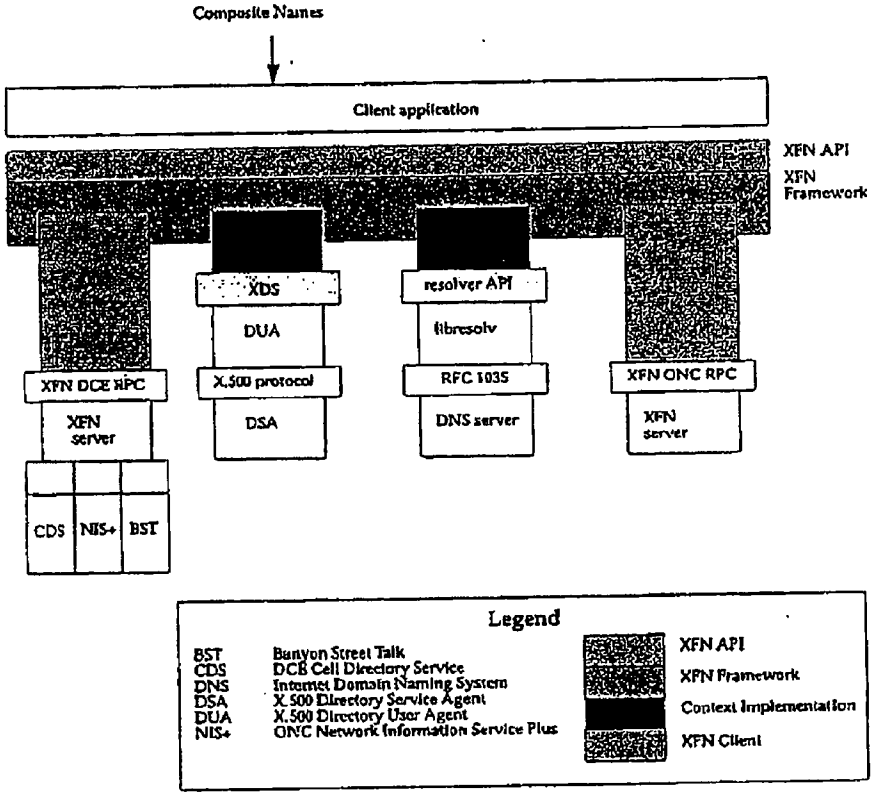


Figure 2-1 Example of an Implementation Model

2.4 Relationship of XFN to other Naming-related Services

2.4.1 Security

XFN does not define a security model nor a common security interface for contexts. Security-related operations, such as those required for authentication or access control, fall into the category of administrative interfaces which are expected to differ widely amongst federation members. Any single choice of security model at the XFN layer is prone to be fundamentally incompatible with the security provided by other direct interfaces, thereby giving rise to inconsistent protection that is susceptible to compromise. Consequently, the security administration interface is kept entirely separate from the federated naming service interface, outside the scope of XFN.

XFN does, however, provide means by which security can be integrated with specific XFN implementations. Operations that fail due to security-related problems can indicate in the status code the nature of the failure.

Authentication

Authentication is to be handled in the modules that implement the service interfaces for each particular member naming system. It occurs as part of the communication that occurs between the client and the naming service. Significant engineering issues remain when multiple security mechanisms and administrative domains are involved. These problems can be addressed on a one-to-one basis, or via a general federated security model, outside the scope of XFN.

XFN provides the status code [FN_E_AUTHENTICATION_FAILURE] to indicate that an operation failed due to authentication errors.

Access Control

Given the ability to authenticate the principal making a service request, a context service provider must then decide whether this principal should be granted or denied the request. Access control is to be handled through additional interfaces in the specific contexts. This can be done by having operations that control default authorisation at context creation and binding creation times, and having interfaces that modify the current authorisation settings. This is analogous to the *umask* and *chmod* scheme for POSIX.1 files.

The access control model is outside the scope of XFN.

XFN provides the status codes [FN_E_CTX_NO_PERMISSION] and [FN_E_ATTR_NO_PERMISSION] to indicate that an operation failed due to access control errors. In the case that the principal is not authorised to know that access has been denied due to permission problems, the status code [FN_E_NAME_NOT_FOUND] or [FN_E_NO_SUCH_ATTRIBUTE] is returned.

Appendix C

Guidelines for Federating a Naming System

These guidelines are intended for naming system implementors who either want to integrate an existing naming system into the federation or develop a new XFN-conformant naming system (including any application that exports naming interfaces).

These guidelines are to be used in conjunction with the native specifications of the naming system to be federated and the implementation specification of the XFN client library.

The first section of this appendix describes some typical implementation models that an implementor can use to provide XFN service.

The remaining sections summarise information contained in Chapter 3 and Chapter 4. These contain guidelines to help integrate a naming system into the XFN federation. It is recommended that integrators publish specifications that define the behaviour of their XFN-conformant naming systems. This provides application writers that use XFN with necessary information for writing portable applications and administrators with the necessary information to integrate that naming system into the federation.

C.1 Implementation Models

These guidelines do not prescribe any particular implementation model but in order to appreciate the features of the different possible configurations of the XFN system, it might be helpful to understand the various building blocks and its required functionality.

The three diagrams Figure C-1, Figure C-2 and Figure C-3 serve as examples of the conceptual models of the different possible configurations. The dark shaded boxes shown in these diagrams are building blocks that a naming service integrator needs to provide in order to integrate the naming system with XFN. The modules depicted in the three diagrams are defined as follows:

XFN API
: The *XFN*API is the complete set of interface operations defined in this XFN specification.

XFN Framework
: The *XFN framework* is the implementation of the XFN API, including the client library and the service provider interfaces necessary for integrating native naming systems.

Context Implementation
: The *context implementation* is the naming service-specific module on the XFN client system that is required to integrate legacy naming systems with XFN. The code of the context implementation is a wrapper that maps the XFN API to the API exported by the legacy naming system. The complexity of the context implementation depends on how well the XFN API maps to the native naming service API and which XFN operations are to be supported. At a minimum, the name resolution phase of all operations must be supported.

  The techniques used to access the naming service-specific context implementations from the XFN framework may vary. For systems that support shared libraries and dynamic linking, a common approach might be that the context implementations are dynamically loadable modules.

  This approach of integrating a naming service using a context implementation module does not require any modification to the existing naming service's source code nor does it require access to the naming service's source code. All that is needed is access to the module (library) that exports the naming service-specific API. This approach is by far the easiest and fastest way of adding an existing naming system into the XFN federation.

XFN Client
: The *XFN client* is a module that implements the client protocol machines for the XFN protocols.

  Two XFN protocols are specified in Appendix A, the RPC based protocols for ONC+ systems (specified in RPCL) and for DCE environments (specified in IDL). The list of supported protocols might evolve over time.

  In addition to supporting the protocols, the XFN client might provide services typically offered by naming clients, such as caching. The extent of this support is implementation-specific.

XFN Protocol Exporter
: The *XFN protocol exporter* is the module required on systems that export one of the XFN protocols. This could be a new naming system, an existing naming system that was modified to also support XFN protocols, or a system that supports the XFN client library and also exports XFN protocols (capable of acting as surrogate client).

  The advantage for naming systems that support one of the specified protocols is that any existing XFN client that imports the protocols can be used to communicate with it. This is particularly useful for applications that need to export naming interfaces. Application

programmers do not have to duplicate the client-side implementation and they do not have to invent new naming interfaces. This provides additional benefits such as the ability to utilise caching and other mechanisms provided by the XFN client implementations, and a direct (and possibly more efficient) mapping of XFN operations to the application's naming operations.

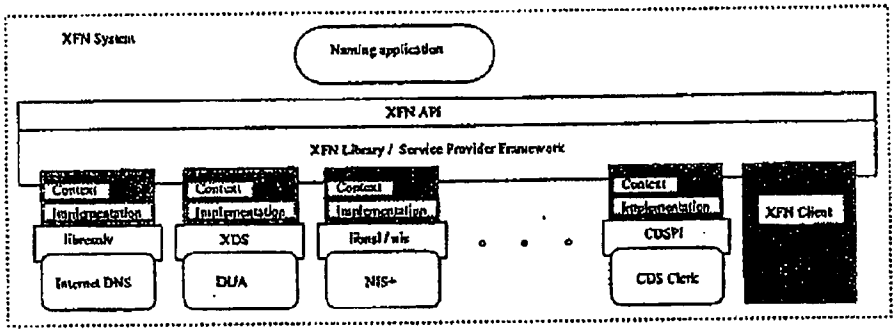


Figure C-1 XFN Configuration using Client Context Implementations

Figure C-1 shows the layering of the XFN client library on top of existing naming system clients on the same system. None of the legacy naming systems needs to be modified.

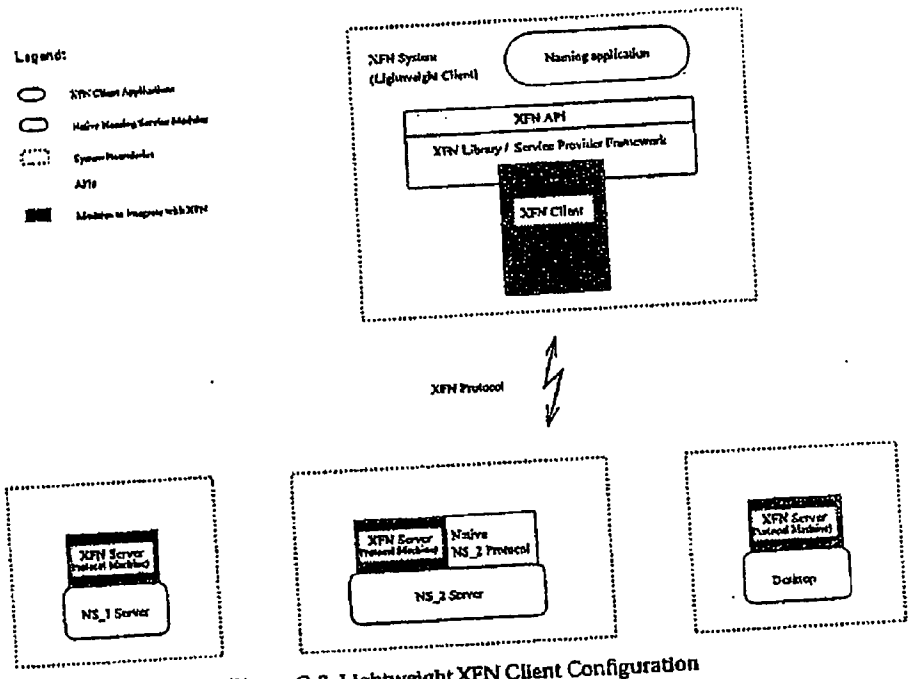


Figure C-2 Lightweight XFN Client Configuration

Figure C-2 shows multiple XFN systems that are connected via one of the specified XFN protocols. The client in this picture is a lightweight XFN client. The servers shown are name servers that directly export one of the specified XFN protocols.

The two modules shown in Figure C-3 are a lightweight XFN client and a server that acts as an intermediary. Similar to the client in Figure C-2, the client in Figure C-3 is a truly lightweight XFN client. None of the legacy naming system clients needs to be installed at that system. Depending on the client system's requirements, the XFN client can be implemented and configured to consume more or less resources, determined based on needs and availability. The XFN client might simply defer to mechanisms (such as for caching and replication) provided by the native naming system clients.

The legacy naming system clients in Figure C-3 reside on a remote system (similar to Figure C-1) that also exports at least one of the XFN protocols. This remote client can be viewed as a surrogate or proxy client that acts on behalf of the initial requestor and performs the native naming system functions.

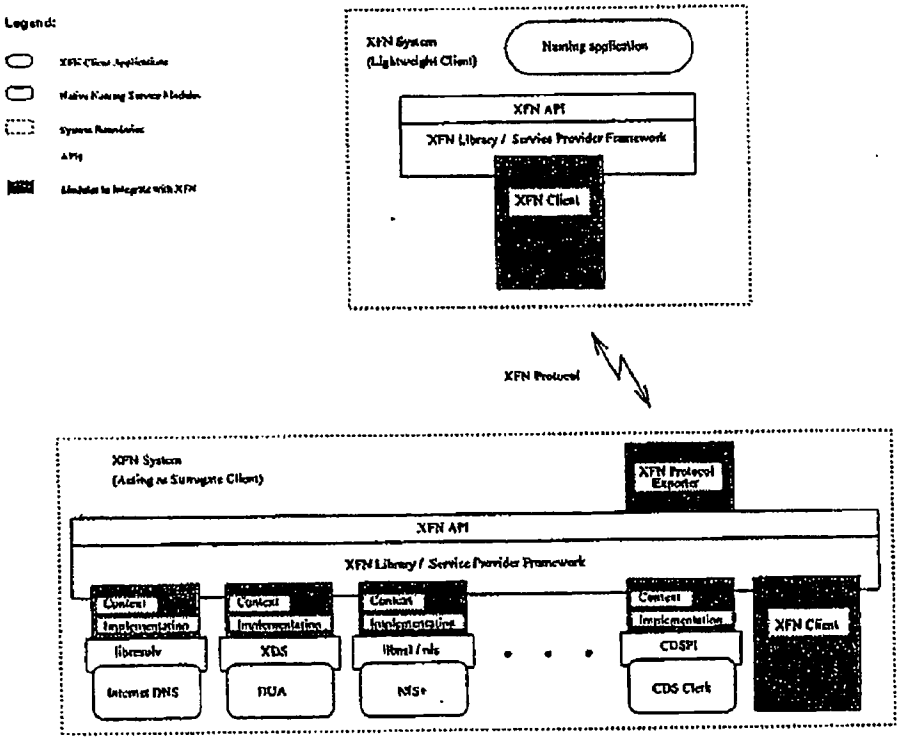


Figure C-3 XFN Configuration with Surrogate Client

Another aspect shown in Figure C-1 and Figure C-3 is the capability of the surrogate client to also import the XFN protocol (*XFN client* module). Such a configuration could serve emerging XFN servers or existing name servers that export one of the specified XFN protocols in addition to, or in replacement of, the native protocol.

Note that a *context implementation* precisely defines the set of modules that are co-located with the XFN framework to map the XFN API to the native naming service API. However, in the context of this appendix, we also use the term *context implementation* to mean the XFN mapping

code that is necessary at the server of a naming system that directly exports one of the XFN protocols (*XFN Protocol Exporter* and *XFN Server* in the diagrams).

C.2 Federating with other Naming Systems

The role and potential configuration of the naming system to be federated must be determined. The naming system might be a global naming system, and therefore should provide some means for registering enterprises and possibly other application naming systems. Or, the naming system could be an enterprise-level or application-specific naming system.

Furthermore, it must be specified whether the naming system must always be joined at the root or whether other entry points are also supported.

Any naming system that participates in the federation must define whether other naming systems can be federated as subordinate naming systems. Naming systems in the application namespaces (such as file systems) might not permit further federation.

Subordinate naming systems are federated through *next naming system* pointers (*nns* pointers). There are two flavours of XFN *nns* pointers: *junctions* and *implicit*. Both may be supported concurrently in a single naming system.

The criteria for whether a naming systems support junctions or implicit *nns* pointers, or both, are in Section 4.3 on page 63. For example, if a distinct name for naming the subordinate naming system is either not available (for example, already in use) or not desired (for example, it would not be a terminal name in that naming system), using implicit *nns* pointers is the preferred technique. An instance where junctions are better suited could be if subordinate naming systems must be selectable by explicit names. One might want to apply the rule of the thumb that implicit *nns* pointers are preferred for the global namespace (because X.500 and DNS names cannot be guaranteed to be leaf names) and most enterprise naming systems are expected to support junctions.

C.2.1 Junctions

Junctions are terminal names in a naming system that hold the reference information of subordinate naming system contexts.

The context implementation defines how references are constructed from information that is associated with the named entry. A common approach might be to use attributes for storing the appropriate reference and address information. Implementations might or might not permit the use of *bind* and *unbind* operations to create and delete references of junctions.

If junctions are supported, the context implementation must ensure the correct behaviour of XFN operations on junctions. In particular, the name resolution phase in XFN operations must follow a junction and resolve to the context that is bound by the reference of the junction.

C.2.2 Implicit Next Naming System Pointers

If the context implementation supports implicit next naming system pointers for federating naming systems, the context implementation must determine how references are managed.

The context implementation must ensure the correct behaviour of XFN operations on implicit *nns* pointers. If a name resolution is performed on a name that has a trailing XFN component separator, the implicit *nns* pointer must be followed.

C.3 Name Syntax

For naming systems participating in the naming federation it is necessary to specify the syntax of the compound name and to specify if the context supports the weak or strong separation model.

C.3.1 Weak and Strong Separation

If the component separator for atomic names of the naming system is distinct from the XFN component separator ('/') and if this XFN component separator character is not used in the compound name in unescaped and unquoted form, only the strong separation model applies.

If the atomic name separator is the same as the XFN component separator and the ordering of atomic names is not left-to-right, strong separation must be enforced by either quoting the compound name or escaping the atomic name separators. It must be specified which rule — quoting or escaping — applies (possibly both).

If the atomic name separator is the same as the XFN component separator and the ordering of atomic names is left-to-right, either strong or weak separation might be supported, or both. Please note, that in the instance where both forms of separation are supported, the context implementation must be prepared to receive in a XFN component name either a full compound name or a single atomic name. Weak separation can only be supported if at least one of the conditions that are specified in Section 4.2 on page 60 apply, namely the naming system must:

- be a terminal naming system
- If not terminal, the context must be able to do a syntax-specific discovery of naming system boundaries
- If not terminal, the context must be able to return the remaining unresolved components.

It must be specified which rule applies. If syntax-specific checking is done, the restriction for the subordinate naming systems must be clearly specified in order to avoid the use of (top level) names that conflict with the syntax rules. For instance, if the name syntax defines typed names for the atomic name component that uses the equal character ('=') as the type separator, the first name component of the subordinate naming system must not contain an unescaped equal character.

C.3.2 Syntax Attributes

In order to permit applications to use the operations on compound names, the context implementation must provide for a means of supporting the interface function for retrieving the syntax attributes (*fn_ctx_get_syntax_attrs*()). If the "fn_syntax_type" attribute indicates the support of the XFN standard model, it can be assumed that the XFN framework implementation provides the support for interpreting the syntax attribute values. If the XFN standard model is not supported, the context implementation must provide the appropriate module for evaluating the syntax attributes and for compound name resolution. (Refer to Section 3.8 on page 50, for more information on the syntax attributes and the XFN standard model).

If the syntax attributes are fixed for a given naming system, the context implementation might not provide interfaces for modifying these attributes; in fact they might not be implemented as regular attributes of the naming system. It should be specified how these attributes are determined and whether they can be set and modified (through XFN attribute operations, for instance).

C.4 Context Operations

A conformance statement for the federated naming system must specify the level of support of the XFN context operations. The minimum requirement for XFN conformance is the support of the *fn_ctx_lookup*() operation and the support of the name resolution phase of the other operations.

Depending on the semantics of operations in the underlying naming system and the desired level of integration, other context operations might also be supported.

Operations that are not supported still must be provided with implementations that perform at least the name resolution phase to the target context. If the target context does not support the actual operation, the implementation returns the status [FN_E_OPERATION_NOT_SUPPORTED].

The context implementation may support operations such as *fn_ctx_list_names*(), *fn_ctx_list_bindings*() and *fn_ctx_create_subcontext*(), but the native naming system might not provide these as atomic operations. The atomicity guarantees for such operations must be specified.

There are a number of context operations whose behaviour depends on the semantics of operations in the underlying naming system. The implementation specifications must specify how the XFN operations effect the state of the underlying naming system. The XFN context operations that might incur different behaviour on different naming systems are detailed in the following subsections (For details on the operation's semantics, refer to the manual reference pages in Chapter 6.):

fn_ctx_bind()
: Naming systems might or might not support the *exclusive* flag.

  Naming systems might or might not permit binding a name without also creating some attributes first. Also, the *fn_ctx_bind*() operation might only permit creation of certain types of objects in the namespace.

  If next naming system pointers are supported, implementations might not permit binding of these references through the *fn_ctx_bind*() operation.

fn_ctx_unbind()
: The failure semantics of this operation is different depending on the type of object in the namespace. For example, an *fn_ctx_unbind*() on some contexts might not be permitted if it is not terminal (for example, if it is a directory).

fn_ctx_create_subcontext()
: Similar to *fn_ctx_bind*(), some naming systems might not permit creating a subcontext without also creating some attributes first.

  If naming systems support namespace partitioning or replicated servers, the *fn_ctx_create_subcontext*() might not provide the sufficient information to perform the creation.

fn_ctx_destroy_subcontext()
: Similar to *fn_ctx_unbind*().

fn_ctx_rename()
: The scope of the *fn_ctx_rename*() operation must be specified. Some naming systems might have restrictions on the semantics of rename. For example, only renames within the same context or on the same server might be permitted.

fn_ctx_lookup_link()
The implementation of XFN links must be specified (the information could possibly be stored in specific attributes). Also, the relationship of XFN links to any support for native soft links or aliases in the underlying naming system needs to be specified.

fn_ctx_handle_from_ref()
The implementation of the XFN framework in conjunction with the different context implementations determine the details of this operation. It is the responsibility of specific context implementations to define the properties of the FN_ctx_t object that is returned and to maintain the context handle and appropriate state information.

fn_ctx_get_ref()
Implementations may vary in their support for this operation in that the returned reference may contain a list of addresses that is different from the originally supplied to *fn_ctx_handle_from_ref*().

C.5 Attribute Operations

Similar to the context operations, a conformance statement for the federated naming system must specify the level of support of the XFN attribute operations.

The operations on attributes might be partially, fully, or not supported. Typically, partial support would contain the operations on single attributes and exclude the multi-attribute operations.

Operations that are not supported still must be provided with implementations that perform at least the name resolution phase to the target context. If the target context does not support the operation, the function returns the status {FN_E_OPERATION_NOT_SUPPORTED}.

The context implementation may support operations such as *fn_attr_get_ids*(), *fn_attr_multi_get*() and *fn_attr_multi_modify*() but the native naming system might not provide these as atomic operations. The atomicity guarantees for such operations must be specified.

If the attribute operations are at least partially supported, the implementation specification of the naming service must specify how its attribute model maps to the XFN attribute operations. Section 3.3.2 on page 27 discusses that the attribute model is not dictated by XFN. Some naming systems might associate attributes with the named objects, others might support attributes that are associated with names in contexts that are bound to objects, or naming systems support a combination of both. Some naming systems might not even distinguish between the two models.

Implementations might only support the attribute operations for certain types of attributes or attributes that are associated with certain type of entries in the namespace (for instance, directories or soft links may be excluded).

For the attribute modify operations, implementations must determine how the operation codes FN_ATTR_OP_ADD, FN_ATTR_OP_ADD_EXCLUSIVE, FN_ATTR_OP_REMOVE, FN_ATTR_OP_ADD_VALUES and FN_ATTR_OP_REMOVE_VALUES apply and what the exact semantics are.

C.5.1 Attributes and Next Naming System Pointers

In order to support next naming system pointers, the implementation specification must define how access to objects that the next naming system pointers is bound to (the contexts of the subordinate naming systems) can be disambiguated from access to attributes in the superior naming system. This disambiguation is particularly necessary for junctions where attributes are maintained in both the name of the junction in the superior naming system and the root context of the subordinate naming system.

If implicit *nns* pointers are used and one wants to access attributes that are associated with the subordinate naming system context, a trailing slash (XFN component separator) used for the resolution of the context disambiguates the access. If a trailing slash is passed in the *name* argument of an attribute operation, an implementation might support operations on attributes that are associated with the implicit *nns* pointer. (The analogy to the behaviour of junctions becomes obvious if one considers a trailing slash as a trailing empty name — this *nns* pointer is implicit).

For junctions, no trailing XFN component separators are permitted, but one typically accesses attributes associated with the subordinate naming system context if the context of the fully resolved name is passed as an argument to the attribute operation (the *name* argument is empty). In contrast, if the name of the junction is passed in the *name* argument, one might access attributes associated with the junction name (this might or might not be supported by naming system implementations).

Figure C-4 and Figure C-5 demonstrate possible associations between attributes, objects and names in a naming system. These diagrams show how one might use attributes to maintain references for next naming system pointers.

In the first example, Figure C-4, the reference of an implicit next naming system pointer is supported in the global namespace as an attribute that is associated with a context object.

If we take the name .../umass.edu/ as an example (both with and without trailing slash), we find that there are four possible ways how this name can be represented to the attribute operations, depending on what the starting context (the object passed to the *ctx* argument) is set to:

ctx of	*name*	# in Diagram	Attribute Operation Refers To
.../edu	umass	(1)	Attributes associated with name umass in the context of edu
.../umass.edu	*empty*	(2)	Attribute associated with the context of *umass*
.../umass.edu	"/"	(3)	Attributes associated with anonymous *nns* pointer maintained in context umass
.../umass.edu/	*empty*	(4)	Attributes associated with root context of next naming system

The second example, Figure C-5, depicts a possible implementation for junctions. The junction's reference is maintained in an attribute (or conceivably set of attributes) that is associated with a terminal name of the enterprise naming system.

If we take the name /.../umass.edu/science/service/spreadsheet as an example for naming a junction, we have two ways of passing this to one of the attribute operations (note that trailing slashes are not permitted for junctions):

ctx of	*name*	# in Diagram	Attribute Operations Refers To
service	spreadsheet	(5)	Attributes associated with junction name spreadsheet in the context of service
service/spreadsheet	*empty*	(6)	Attributes associated with root context of next naming system

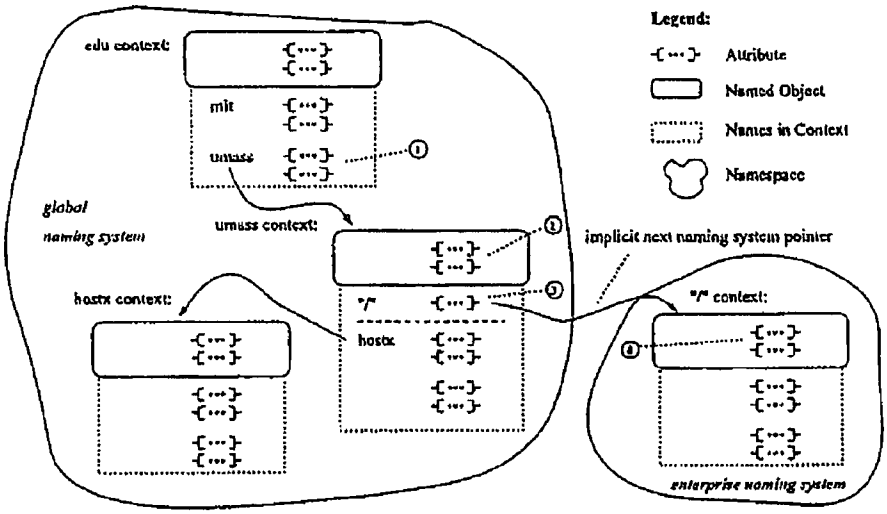


Figure C-4 Attribute Example with Implicit Next Naming System Pointer

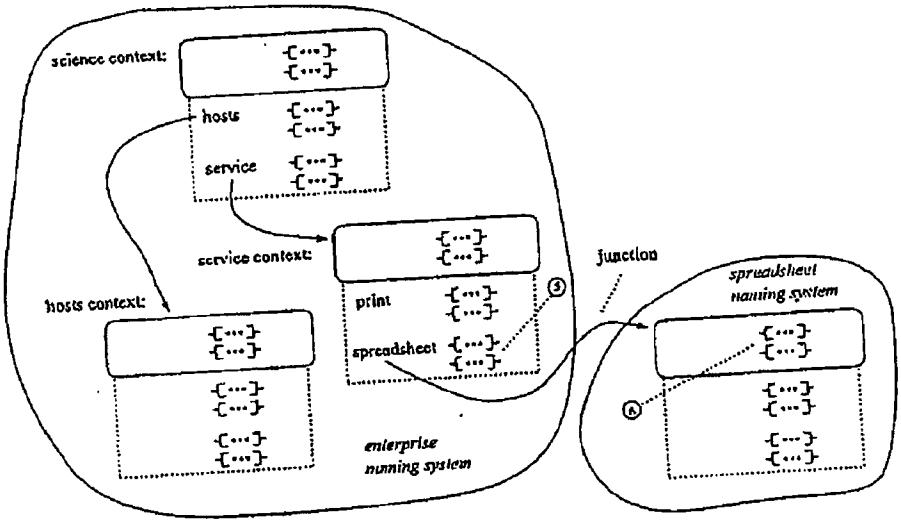


Figure C-5 Attribute Example with Junction

C.6 Reference and Address Types and its Registration

The values and encodings of reference and address types exported by the naming system need to be specified in order to support its federation by other naming systems. The implementor of the context implementation must decide which of these types are necessary to be registered with X/Open.

Generally, a single distinct reference type is specified for uniquely identifying a particular naming service. This reference type is used by the XFN framework as the discriminator for locating and dispatching to the context implementation.

XFN framework implementations may also permit default fallbacks where reference type specific implementations are not available. The address types can be used as discriminator instead. If a reference contains multiple address types, the order of selection is unspecified.

In cases where servers of a naming system directly export one of the XFN protocols, the appropriate reference types (for example, FN_DCE_SERVICE_REF) are used for selecting the client implementation. If implementations for the reference types are available, the XFN framework dispatches to the XFN client module that imports these protocols.

It might also be relevant for users and administrators of a federated naming system to know how XFN references and addresses map to its representation in the naming system. Since these might typically be stored in attributes, it is necessary for implementors of a federated naming system to define the format and encodings.